



UNICEUB – CENTRO UNIVERSITÁRIO DE BRASÍLIA

FATECS – FACULDADE DE TECNOLOGIA E CIÊNCIAS SOCIAIS APLICADAS

CURSO DE ENGENHARIA DA COMPUTAÇÃO

BRUNO MOREIRA MATOS

JANELA RESIDENCIAL AUTOMATIZADA

BRASÍLIA / DF

2º SEMESTRE DE 2009

BRUNO MOREIRA MATOS

JANELA RESIDENCIAL AUTOMATIZADA

Monografia apresentada ao Curso de Engenharia da Computação, como requisito parcial para obtenção do grau de Engenheiro de Computação.

Orientador: José Julimá Bezerra Júnior

BRASÍLIA / DF

2º SEMESTRE 2009

Dedico este projeto e monografia, a meus pais, meus irmãos, todos os meus familiares, a minha namorada e todos aqueles que contribuíram diretamente e indiretamente para a elaboração e conclusão desta.

AGRADECIMENTO

Agradeço primeiramente a Deus, por me abençoar com a sabedoria e ,
entendimento, a meus pais pelos esforços incansáveis para formação de meu
caráter, a minha namorada pelo apoio constante, aos meus irmãos, meu sobrinho e
todos os meus familiares, e em caráter especial aos meus professores do qual
compartilharam de conhecimento que me acompanharão por toda a minha jornada
profissional.

O meu muito obrigado!

RESUMO

Este projeto tem a proposta de automatizar janelas residenciais. A maneira escolhida para essa automação foi o controle de um motor de passo por meio de um microcontrolador, quando acionado por um sensor de chuva. O projeto tem a função de abrir a janela com a ausência de chuva e fechar a janela com a presença de chuva. Também foi desenvolvida uma maneira de acionar a abertura e o fechamento da janela por meio de botões, mas como a intenção deste trabalho é não permitir que o interior da residência molhe, os botões de abertura e fechamento da janela só funcionarão quando não houver chuva, impedindo assim que a janela seja aberta com a presença de chuva. Para o funcionamento do sistema é utilizado um motor de passo do tipo híbrido e bipolar que está controlado por um microcontrolador modelo AT89S59, que faz parte da família 8051, fabricado pela Atmel, que também possui a função de receber a informação do usuário.

O sensor de chuva utilizado neste projeto foi construído de forma artesanal, assim o custo do projeto não fica tão alto, para desenvolvê-lo foram utilizados alguns princípios físicos, descritos ao longo desta monografia.

Palavras Chaves: motor de passo, microcontrolador, sensor de chuva, automação.

ABSTRACT

This Project has the proposal to automatize residential window. The way chosen for this automation was the control of an engine of step by means of a microcontroller, when set in motion for a rain sensor. The project has the function to open the window with the rain absence and to close the window with the rain presence. Also a way was developed to set in motion the opening and the closing of the window by means of buttons, but as the intention of this work is not to allow that the interior of the residence has wet, the opening button of the window alone will function when it will not have rain, not alluded the window opened in the rain presence. For the operation of the system a step motor do tipo bipolar will be used that will be controlled for a microcontroller, model AT89S59, of the family 8051, make by Atmel Inc. that also will have the function to receive the information from the user.

The rain sensor utilizing in this project was made by artisanal form, make by this way the project was not expensive. Was utilized some physical principium to developing the project, and they are written in this monographic.

Keywords: step motor, microcontroller, automation rain sensor.

Sumário

AGRADECIMENTO	iv
RESUMO	v
ABSTRACT	vi
LISTA DE FIGURAS	3
LISTA DE QUADROS	5
CAPÍTULO 1 – INTRODUÇÃO	6
1.1. – Apresentação do Problema	6
1.2. – Motivação.....	7
1.3. – Objetivos do Trabalho	7
1.4. – Justificativa e Relevância do Tema	9
1.5. – Resultados do Trabalho	9
1.6. – Estrutura do Trabalho.....	10
CAPÍTULO 2 – REFERENCIAL TECNOLÓGICO.....	12
2.1. – Motor de Passo.....	12
2.1.1. – Tipos básicos.....	15
2.1.2. – Modos de Funcionamento.....	18
2.1.3. – Formas de Operação.....	19
2.1.4. – Motor utilizado no Projeto	20
2.1.5. – Ponte H.....	21
2.2. – Sensor de chuva.....	23
2.2.1. – Dielétrico	24
2.3. – Microcontrolador	27
2.4. – A Família 8051	28
2.4.1. – Pinagem do 8051	30
2.4.2. – Interrupção	34
2.5. – Linguagem de Programação Assembly	35
2.6. – Linguagem de Programação C	36
2.7. – Circuitos Integrados e Componentes.....	37
2.7.1. – Display	37
2.6.2. – L298N	39

CAPÍTULO 3 – SISTEMA DE AUTOMAÇÃO DE JANELAS.....	41
3.1. – Apresentação Geral	41
3.1.1. – Fluxograma Geral.....	42
3.1.2. – Visão geral do Projeto.....	44
3.2. – Controle do Motor de Passo	45
3.3. – Controle do sensor de chuva	47
3.4. – Controle do Display de LCD	48
3.5. – Controle do Microcontrolador.....	49
3.5.1. – Programação de controle	49
3.5.2. – Inserção do algoritmo no Microcontrolador.....	51
3.6. – Apresentação final do circuito	53
3.7. – Apresentação do Protótipo	63
CAPÍTULO 4 – CONSIDERAÇÕES FINAIS	70
4.1. – Dificuldades encontradas	70
4.2. – Avaliação Global do Projeto	71
4.3. – Conclusão	71
4.4. – Sugestões para trabalhos futuros.....	72
REFERENCIAS BIBLIOGRAFICAS	73
APÊNDICE	76
Programa inserido no microcontrolador	76
ANEXO I - DATASHEET DO CI L298N	92
ANEXO II - DATASHEET DO MICROCONTROLADOR AT89S52	96
ANEXO III - DATASHEET DO DISPLAY 16X2	102

LISTA DE FIGURAS

Figura 1 – Esboço do Projeto.....	8
Figura 2 – Motor de Passo.....	12
Figura 3 – Eletroímã	14
Figura 4 – Motor de passo de Relutância Variável	15
Figura 5 – Motor de passo de Imã Permanente	16
Figura 6 – Motor de passo Híbrido	17
Figura 7 – Motor de passo Unipolar	20
Figura 8 – Motor de passo Bipolar	20
Figura 9 – Exemplo de Ponte H	22
Figura 10 – Sensor de Chuva	23
Figura 11 - Microcontrolador	28
Figura 12 – Pinagem do Microcontrolador.....	30
Figura 13 - Display	37
Figura 14 – Pinagem do CI L298N.....	39
Figura 15 – Diagrama de Blocos L298.....	40
Figura 16 – Fluxograma do sistema.....	43
Figura 17 – Visão Geral do Projeto	44
Figura 18 – Programa ConTEXT	51
Figura 19 – Extensões geradas pelo ConTEXT	52
Figura 20 - Gravador de Microcontroladores AT89s51/s52/s53/8252/8253 ATMEL	53
Figura 21 – Ligações do Projeto.....	54
Figura 22 – LCD - Tempo Limpo (a); Abrindo Janela	57
Figura 23 – LCD - Tempo Limpo (a); Janela Aberta	58
Figura 24 – LCD - Esta Chovendo (a); Fechando Janela	59

Figura 25 – LCD - Esta Chovendo (a) Janela Fechada.....	60
Figura 26 – LCD - Tempo Limpo (m); Janela Aberta.....	61
Figura 27 – LCD - Tempo Limpo (m); Janela Fechada	62
Figura 28 - Foto do Protótipo (1).....	63
Figura 29 - Foto do Protótipo (2).....	64
Figura 30 - Foto do Protótipo (3).....	65
Figura 31 - Foto do Protótipo (4).....	66
Figura 32 - Foto do Protótipo (5).....	67
Figura 33 - Foto do Protótipo (6).....	68
Figura 34 - Foto do Protótipo (7).....	69

LISTA DE QUADROS

Quadro 1 – Pinagem do Display	38
Quadro 2 – Comandos do Display	39
Quadro 3 – Possibilidades do sistema	42
Quadro 4 – Passos do motor	46

CAPÍTULO 1 – INTRODUÇÃO

1.1. – Apresentação do Problema

Visando acabar com o problema em residências decorrente das chuvas, há várias empresas no mercado que disponibilizam automações de janelas. A maioria dessas empresas oferece além de automação com sensor de chuva, oferecem um controle remoto para abri-la e fechá-la, outras também oferecem sensor de luz.

Algumas empresas como a Euro Centro Abitare Com Stile (página na web: <http://www.eurocentro.com.br/>) oferecem este tipo de serviço, porém os preços não são tão acessíveis. Conforme o site desta empresa, para uma janela com 55 centímetros por 98 centímetros com motor, sairia por mais de quatro mil reais.

Este trabalho utiliza sensor de chuva, sendo assim, a janela fechará e abrirá com a presença ou não de chuva. O projeto conta também, com controles para abrir e fechar a janela de acordo com a ação do usuário.

Possivelmente em um futuro próximo, os sistemas automatizados estarão presentes em várias residências, trazendo economia, conforto e segurança para as pessoas.

Como este projeto depende das condições climáticas, é muito importante ter componentes que satisfaça o projeto, sendo assim, para o bom funcionamento deste é necessário um bom sensor de chuva.

Este projeto tem a finalidade de automatizar uma janela residencial utilizando recursos mais baratos, melhorando assim o custo benefício do sistema.

1.2. – Motivação

A principal motivação para realização deste projeto é utilizar conhecimentos adquiridos no curso de Engenharia de Computação que são ligadas a área de automação. Várias disciplinas como “Microprocessadores e Microcontroladores”, “Linguagem de Programação”, “Circuitos Eletrônicos”, dentre outras do curso de Engenharia de Computação, fornece uma real percepção de como são projetado os sistemas de automação de uma forma simplificada e com baixo custo, sendo também estudado ao longo do curso o funcionamento do microcontrolador 8051, componente este, utilizado no projeto.

A construção de um sistema de automação residencial de baixo custo foi outra grande motivação para a realização deste projeto, pois as empresas que oferecem este tipo de sistema, geralmente cobram preços não acessíveis a todos.

1.3. – Objetivos do Trabalho

O objetivo deste projeto é automatizar janelas residenciais no controle sobre precipitação ou ausência de chuva, sendo que a janela é aberta ou fechada automaticamente, se não houver ou houver chuva, respectivamente.

Para que seja realizado o objetivo proposto, é necessário utilizar um microcontrolador, que aciona um motor de passo, fazendo com que a janela abra ou feche de acordo com a água atuante em um sensor de chuva.

Para uma maior interação e comodidade do usuário com o sistema, o projeto contará com botões, a fim de ter um controle de abertura e fechamento da janela enquanto não estiver chovendo.

Para interpretar os comandos de abertura e fechamento das janelas e transformá-los em informações capazes de movimentar o motor de passo, é utilizado um microcontrolador da família 8051. Este microcontrolador também é responsável por enviar a informação correta que deve ser mostrada no display, que tem a função de mostrar em sua tela se está chovendo ou não, e se a janela esta aberta ou não.

A Figura 1 mostra em uma visão geral do projeto.

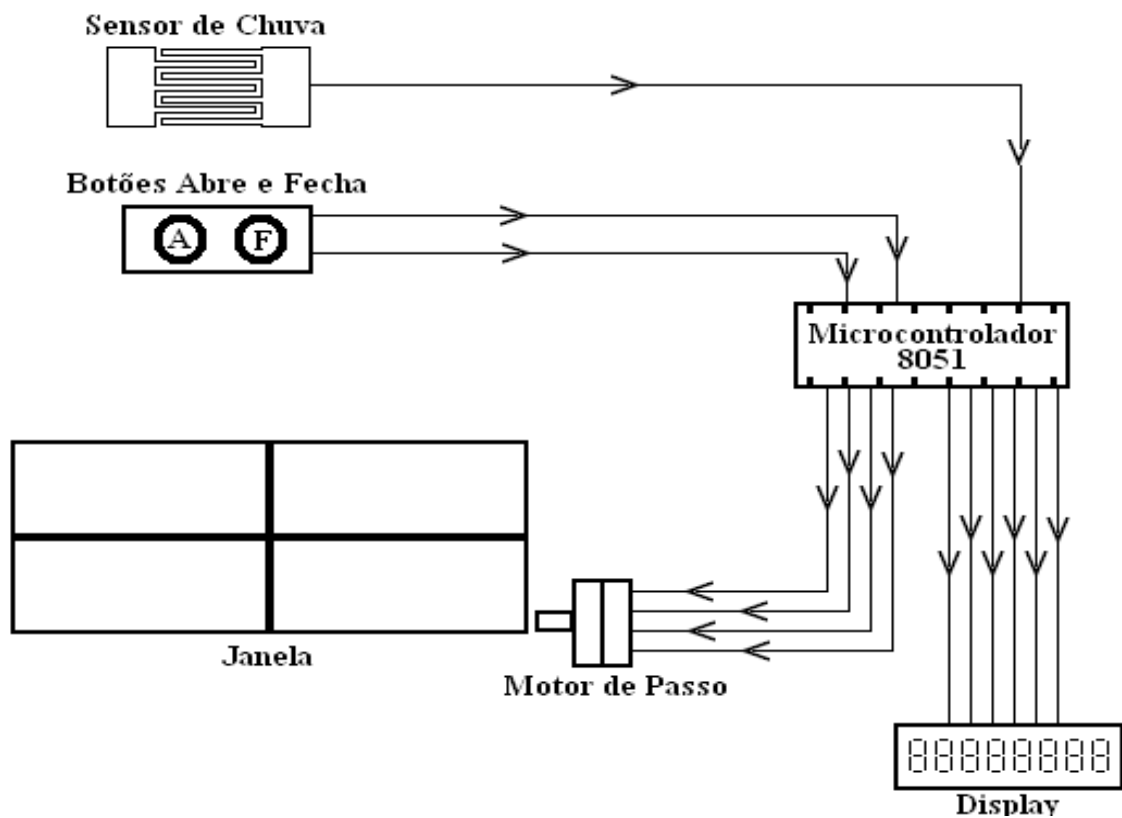


Figura 1 – Esboço do Projeto

(Autor)

1.4. – Justificativa e Relevância do Tema

Este projeto enquadra-se dentro das matérias estudadas no curso de Engenharia de Computação, algumas disciplinas ligadas diretamente ao tema proposto são: Microcontroladores e Microprocessadores, Circuitos e Máquinas Elétricas, Circuitos Eletrônicos, Arquitetura de Computadores, Linguagem e Técnicas de Programação.

O tema proposto é muito atual, sendo que as pesquisas na área de automação estão em ascensão, tanto que, segundo a Associação Brasileira de Automação Residencial, no ano de 2006 houve empresas desta área que tiveram um crescimento de 200%, e a procura dos consumidores por empresas de automação residencial neste ano de 2009, teve um aumento de 30% no primeiro semestre em comparação com o mesmo período do ano anterior, segundo a Associação Brasileira de Automação Residencial (página na web: <http://www.aureside.org.br/>).

O avanço da tecnologia, e facilidades à mão de todos, fazem com que a concorrência nesta área da Engenharia seja cada vez maior, onde não só as grandes empresas têm a possibilidade de elaborar projetos de automação, mas qualquer pessoa que tenha o conhecimento das tecnologias já existentes no mercado.

1.5. – Resultados do Trabalho

Este projeto tem a finalidade construir dois circuitos, sendo um para a automação propriamente dita do motor de passo, e o outro para a interface.

O primeiro circuito tem um microcontrolador ligado ao sensor de chuva, quando houver a presença de chuva o sensor fecha o curto e aciona a lógica do microcontrolador para que a janela seja fechada. Quando o sensor secar (não houver mais a presença de chuva), o microcontrolador deve acionar o motor de passo para que a janela se abra. Paralelamente a este circuito, são inseridos botões para abrir e fechar a janela, que são funcionais enquanto não houver chuva.

O segundo circuito é responsável por manter uma interface, ele deve mostrar na tela de um display se esta chovendo ou não, e se a janela esta aberta ou não. Para este circuito o display é ligado ao microcontrolador que mostra se a janela esta aberta ou fechada, e se esta chovendo ou não.

1.6. – Estrutura do Trabalho

A estrutura desta monografia consiste em quatro capítulos que trata os assuntos descritos abaixo:

Capítulo 1 – É o capítulo introdutório, nele é apresentada à idéia geral do projeto, mostrando os objetivos a serem alcançados, é mostrada uma visão geral de como é desenvolvida a monografia e implementação do projeto.

Capítulo 2 – É o capítulo de referencial tecnológico, onde a teoria utilizada no projeto é demonstrada, abordando uma teoria acerca dos dispositivos utilizados no desenvolvimento do projeto, visando auxiliar e levantar um conhecimento já concretizado para uma melhor compreensão do projeto.

Capítulo 3 – É o capítulo demonstrativo da implementação de software e hardware. Aqui é descrita a conexão entre os componentes do hardware, programação efetuada no microcontrolador e utilização de alguns componentes eletrônicos necessários para funcionamento dos componentes principais ou somente para regulagem, também é mostrando os resultados através de figuras com o funcionamento do protótipo.

Capítulo 4 – É o capítulo onde é deixada a impressão causada pelo projeto, por meio da descrição das dificuldades encontradas e sua avaliação global. Onde as considerações finais serão expostas e descritas as sugestões para trabalhos futuros.

Também faz parte da estrutura desta monografia, um referencial bibliográfico, um apêndice com a programação utilizada, e alguns anexos com datasheet de componentes utilizados.

CAPÍTULO 2 – REFERENCIAL TECNOLÓGICO

Este capítulo tem como objetivo expor um pequeno referencial teórico para o projeto final. As principais teorias relevantes são expostas aqui.

2.1. – Motor de Passo

O motor de passo é um tipo de motor elétrico, isto é, ele é uma máquina que transforma energia elétrica em energia mecânica.

Os motores de passo são projetados a fim de se ter um maior controle sobre a rotação do motor, uma importante característica desses motores, é sua compatibilidade com sistemas digitais. [TORO, 1994]

Na Figura 2 são mostradas duas fotos em ângulos diferentes do motor de passo utilizado no projeto.



Figura 2 – Motor de Passo

(Autor)

Os motores de passo são comumente utilizados em sistemas de controle digital, onde ele recebe comandos na forma de uma sequência de pulsos para girar

ou mover um objeto por uma distancia precisa. Hoje em dia esse tipo de motor é muito encontrado em impressoras, para alimentação de papel e alinhamento do cabeçote de impressão, também vem sendo bastante utilizado em máquinas cirúrgicas, onde os médicos operam os pacientes por meio de máquinas controladas a distância, diminuindo a probabilidade de erro com a precisão dos movimentos, e diminuindo sensivelmente o risco de infecção, uma vez que utilizando essa técnica as incisões são menores. [TORO, 1994]

O motor de passo é largamente utilizado nas aplicações mencionadas por ser um mecanismo que converte informações digitais em uma saída mecânica, tornando-se uma interface satisfatória para computadores digitais e microcontroladores. [TORO, 1994]

As características básicas de um motor de passo consistem em um rotor e um estator. O rotor é a parte interna e móvel do motor, é a parte que gira, já o estator é a parte externa do motor que é estática, é nesta segunda estrutura que são enroladas as bobinas, enrolamentos de fios de cobre, que criam o campo magnético girante. O conjunto de estator e rotor estão separados por um pequeno espaço vazio entre eles, para que seja possível o movimento de rotação.

A disposição dos enrolamentos de fios nas bobinas do motor de passo gera um efeito semelhante àquelas encontradas nos ímãs naturais. Este efeito é criado por um dispositivo que utiliza corrente elétrica para gerar um campo magnético, conhecido como eletroímã. Ele geralmente é construído aplicando-se um fio elétrico espiralado ao redor de um núcleo de ferro, aço, níquel ou cobalto. A Figura representa um eletroímã. [UNIMAR]

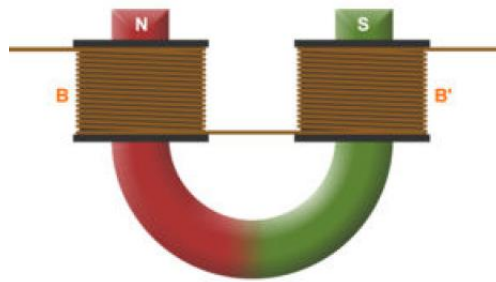


Figura 3 – Eletroímã

[UNIMAR]

Quando o fio é percorrido por uma corrente elétrica, a espira se submete a uma tensão, o que gerará um campo magnético na área circunvizinha a essa espira (a intensidade do campo e a distância que ele atingirá a partir do eletroímã dependerão da intensidade da corrente aplicada e do número de voltas da espira) através da Lei de Biot-Savart. [UNIMAR]

A passagem de corrente elétrica por um condutor produz campos magnéticos nas imediações do condutor e estabelece-se um fluxo magnético no material ferromagnético envolto pelas espiras do condutor, a razão entre a intensidade do fluxo magnético concatenado pelas espiras e a corrente que produziu esse fluxo é a indutância, a intensidade do campo magnético produzido dependerá das características do meio no qual se estabelecerá o fluxo, o que afetará a relutância do caminho magnético. [UNIMAR]

Com o motor de passo é possível ter controle sobre o fluxo de corrente nas espiras das bobinas, sendo assim é possível ter o controle da polaridade de cada uma, podendo girar o motor conforme necessidade.

2.1.1. – Tipos básicos

Os motores de passo são classificados de acordo com as características de sua construção, eles são divididos em três tipos principais, o de Relutância Variável, o de Imã Permanente, e o Híbrido. [Fitzgerald, 2006]

Um motor de passo de Relutância Variável tem como característica um estator laminado e um rotor disposto com várias polaridades desenvolvido com ferro doce, que em sua maioria possui passos altos, na faixa de 5 a 15 graus. Como não possuem ímãs apresentam torque estático nulo. A Figura 3 representa o funcionamento de um motor de passo de Relutância Variável.

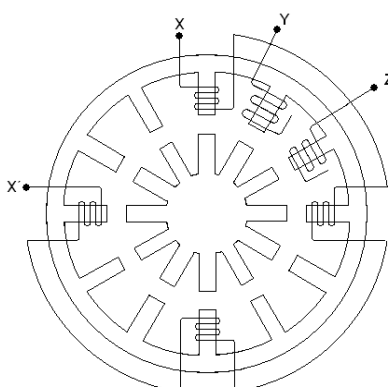


Figura 4 – Motor de passo de Relutância Variável

[Fitzgerald, 2006]

Quando a fase X for energizada, os quatro dentes do estator, que estão com a fiação da fase X, se alinharão com os quatro dentes do rotor por meio de atração magnética, neste ponto o motor estará parado, para que ele dê o primeiro passo a fase X deverá ser desligada e a fase Y deverá ser ligada, assim os quatro dentes do rotor se alinharão com os dentes do estator e o motor terá dado o um passo no sentido horário, no caso da Figura 3 o passo será de 15 graus. Para que o motor

continue girando, é só desligar a fase Y e energizar a fase Z, depois energizar novamente a fase X, e assim sucessivamente. [Fitzgerald, 2006]

Outro tipo de motor de passo é o de Imã Permanente, Figura 4, esses motores se caracterizam por possuírem rotores sem dentes e magnetizados perpendicularmente ao eixo. São construídos com materiais alcalinos ou ferrites, e devido à sua magnetização perpendicular possuem torque estático não nulo. A Figura 4 representa o funcionamento do motor de passo de Imã Permanente.

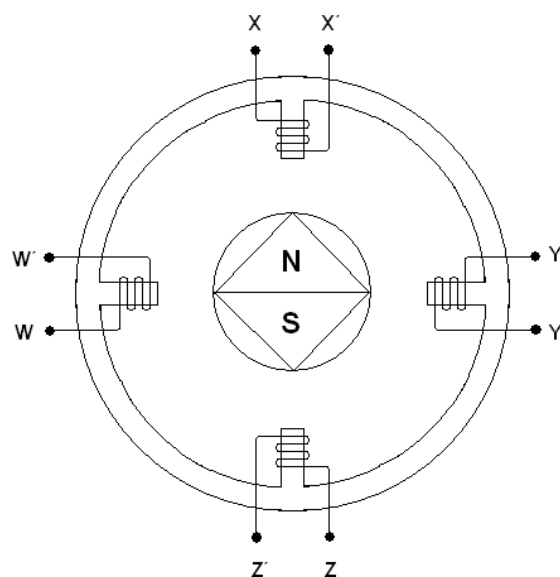


Figura 5 – Motor de passo de Imã Permanente

[Fitzgerald, 2006]

Quando energizamos a fase X o rotor aponta para cima, quando energizarmos a fase Y, o rotor apontará para a direita, fazendo com que o motor gire em um passo de 45 graus no sentido horário. Para o motor continuar a girar no sentido horário, é necessário energizar as fases XYZW seqüencialmente. Os motores de Imã Permanente possuem na maioria das vezes taxas de passo

relativamente baixas, que ficam na ordem de 45 ou 90 graus, em contrapartida oferecem um torque mais elevado. [Fitzgerald, 2006]

O terceiro tipo de motor de passo é o Híbrido, este tipo de motor possui um combinado das características dos motores de Relutância Variável e dos de Imã Permanente, sendo que possuem somente as características desejáveis destes dois tipos. Seus principais atributos consistem em ter um alto torque, ter o torque estático não nulo tem o poder de operar com altas velocidades de passo, e possuem ângulos de passo pequenos que ficam na casa entre 0,9 a 5 graus por passo. Na maioria dos casos, são formados por pólos conseguidos através de dois enrolamentos, conforme Figura 5, desta forma é possível à utilização de apenas uma fonte de alimentação.

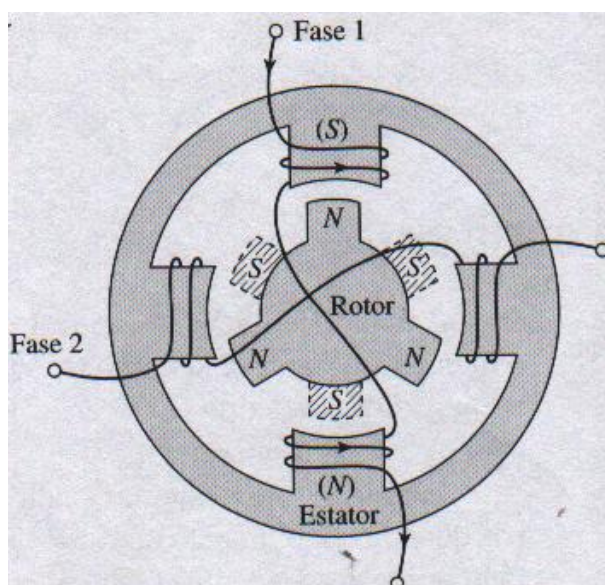


Figura 6 – Motor de passo Híbrido

[Fitzgerald, 2006]

A Figura 5 representa um motor de passo híbrido onde a fase 1 está energizada, de tal forma que a parte superior do estator seja um pólo sul e a parte inferior seja um pólo norte, esta configuração faz com que o rotor, constituído de Imã

Permanente, se alinhe com o estator conforme a Figura 5. Para que o motor dê passos, e conseqüentemente gire, deve-se retirar a energia da fase 1 e energizar a fase 2, de tal forma que o lado esquerdo do estator seja o pólo sul e o lado direito seja o pólo norte, assim o motor girará 30 graus no sentido horário, é importante observar que se as polaridades do estator fossem trocadas o motor giraria no sentido anti-horário. Para continuar a girar, deve-se desligar a fase 2 e energizar a fase 1, porém os pólos das fases devem estar invertidos em relação à situação inicial, e assim seqüencialmente. [Fitzgerald, 2006]

2.1.2. – Modos de Funcionamento

Os motores de passo também podem ser usados de três modos distintos, essa variação esta relacionada ao número de passos por revolução que o motor faz, e ao modo de energização utilizado. [Brites, 2008]

O primeiro modo é o passo normal (*Full-Step*), nesta operação é utilizado o passo normal do motor. A utilização deste modo pode ser feita de duas formas, na primeira forma o motor funciona com uma única fase energizada por vez. Esta forma é indicada quando não é importante nem o torque nem a velocidade, é requerido uma menor quantidade de potência. A segunda forma do passo normal acontece quando energizamos duas fases por vez, dessa forma diminuimos o problema com o torque e com a velocidade, porém é necessário o dobro de potência para o funcionamento do motor, em relação à utilização de apenas uma fase.

O segundo modo de energização de um motor de passo é o Meio-Passo (*Half-Step*), este modo consiste na alternância da energização de uma fase e

energização de duas fases ao mesmo tempo, resultando em passos com a metade do tamanho de um passo normal, aumentando assim a precisão do motor. O torque do motor, neste caso, varia de acordo com o número de fases energizadas. Com essa disposição, este motor se encontra livre de ressonância, podendo utilizar uma grande faixa de velocidade e de carga.

O terceiro e último modo é o Micro-Passo (*Micro-Step*), neste modo o passo natural de um motor pode ser dividido em outros vários passos menores. Os motores de Micro-Passos são produzidos adicionando nas duas bobinas correntes de acordo com o seno e co-seno. Este modo proporciona uma resolução maior, e é utilizado quando há necessidade de movimentos sem trancos e de muita precisão.

2.1.3. – Formas de Operação

Os motores de passo podem ser classificados como bipolar ou unipolar, dependendo do número de enrolamento por fase do motor. [Brites, 2008]

O motor de passo unipolar, Figura 6, tem dois enrolamentos por fase, sendo um para cada sentido da corrente. Desta forma sua configuração se torna mais simples de trabalhar já que cada bobina recebe o sinal independente, sem ter que se preocupar com sentido da corrente.

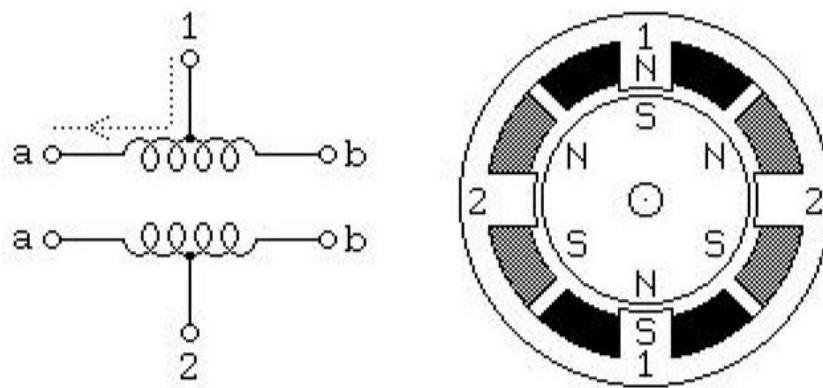


Figura 7 – Motor de passo Unipolar

[Brites, 2008]

O motor de passo bipolar, Figura 8, tem um único enrolamento por fase. Sendo assim, a corrente de um enrolamento necessita ser invertida para que inverta o pólo magnético, com isso seu circuito é um pouco mais complicado que o circuito do motor de passo unipolar. Este projeto utiliza um motor do tipo híbrido e bipolar.

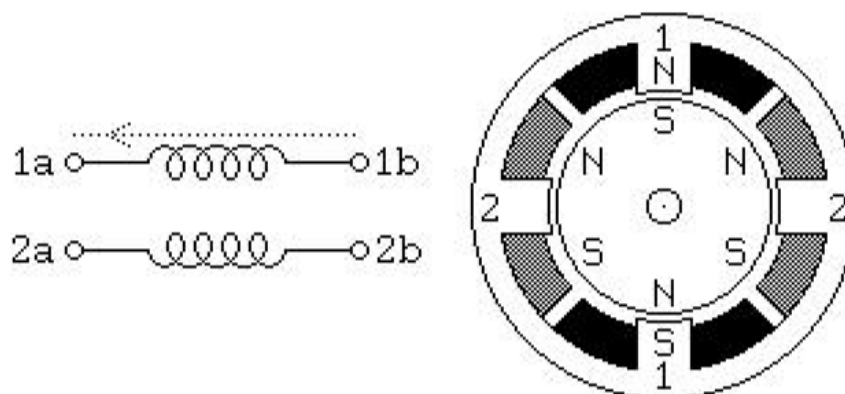


Figura 8 – Motor de passo Bipolar

[Brites, 2008]

2.1.4. – Motor utilizado no Projeto

O motor de passo utilizado no projeto foi o motor de passo híbrido que opera de forma bipolar, no protótipo é utilizado o passo normal do motor (*Full-Step*).

O principal motivo de escolher um motor de passo híbrido se dá pelas suas características, ele possui um alto torque, sendo que o torque estático dele é não nulo, pode operar com velocidade elevada nos passos e este tipo de motor possui uma precisão maior em seus passos.

A forma de operação bipolar foi a única encontrada no mercado para compra do motor de passo, a operação da forma unipolar tem a manipulação mais fácil, porém, com alguns componentes também pode-se utilizar a forma bipolar sem maiores problemas.

O modo de funcionamento dos passos do motor utilizada no protótipo é o passo normal (*Full-Step*), este é o modo mais simples de passos do motor e é utilizado por satisfazer as exigências do protótipo.

2.1.5. – Ponte H

Ponte H é um circuito eletrônico que permite que um motor rode tanto para um sentido quanto para o outro. Estes circuitos são geralmente utilizados em robótica e estão disponíveis em circuitos prontos ou podem ser construídos por componentes.

A Figura 8 mostra um circuito elétrico que é um exemplo de ponte H.

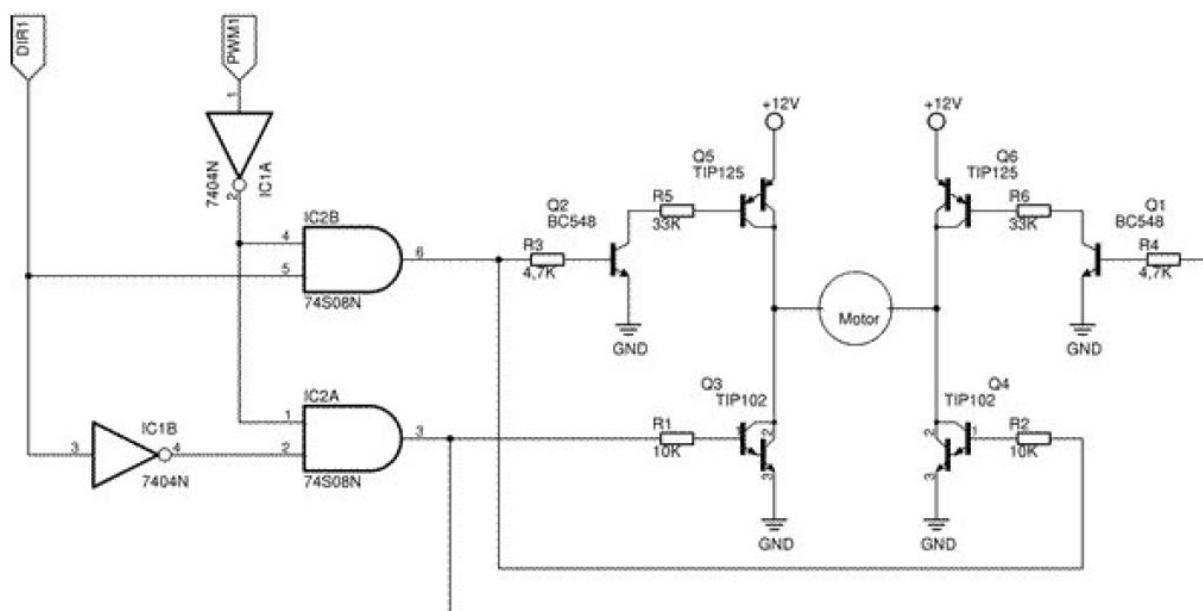


Figura 9 – Exemplo de Ponte H

[Brites, 2008]

O nome ponte H é dado pela forma que assume o circuito quando montado. A Figura 8 mostra um circuito construído com quatro “chaves” (S1-S4) que são acionadas de forma alternada (S1 e S4 ou S2 e S3). Para cada configuração das chaves o motor gira em um sentido. As chaves S1 e S2 assim como as chaves S3 e S4 não podem ser ligadas ao mesmo tempo, pois podem gerar um curto circuito. Para construção da ponte H pode ser utilizado qualquer tipo de componente que simule uma chave liga-desliga como transistores e relés. Para que o circuito fique protegido, é aconselhável que sejam configuradas portas lógicas com componentes 7408 e 7406 a fim de que nunca ocorram as situações de curto circuito descritas acima. Outro melhoramento que pode ser feito à ponte H, seria a colocação de diodos entre as “chaves”, pois quando a corrente não tem onde circular, no caso de o motor parar, ela volta para a fonte de alimentação economizando assim o gasto de energia de uma bateria por exemplo. [Brites, 2008]

2.2. – Sensor de chuva

Este projeto conta com um sensor de chuva projetado artesanalmente, conforme pode ser verificado na Figura 9. Devido à grande dificuldade de encontrar componentes no mercado que atenda às necessidades deste projeto, e devido ao intuito do projeto de ter sempre um melhor custo benefício, este procedimento é o mais viável.



Figura 10 – Sensor de Chuva

(Autor)

O sensor de chuva tem um funcionamento simples de ser entendido. As duas placas de alumínio (Figura 9) são coladas de forma que não se encostem, sendo assim, quando se liga uma corrente positiva de um lado e negativa do outro nada ocorre, pois a distância existente entre as placas de alumínio não deixa a corrente elétrica passar de um lado para o outro, deixando o circuito aberto. Porém, quando um pingo de chuva cai sobre o intervalo entre as placas, ocorre o contato entre elas através da água, fechando assim o circuito.

Vários testes foram realizados para garantir o funcionamento do sensor de chuva criado, de acordo com eles o sensor de chuva não é acionado caso a

umidade relativa do ar esteja próximo dos 100%, porém caso haja sereno forte que molhe o sensor a ponto de criar gotas, aí sim ele será acionado. De uma forma geral o sensor de chuva se mostrou muito satisfatório para o projeto.

2.2.1. – Dielétrico

Todo material não condutor, como ar, vidro, papel ou madeira, é chamado de dielétrico. O sensor de chuva tem um funcionamento similar a de um capacitor, que tem como definição ser um sistema constituído por dois condutores separados por um isolante. No caso do sensor de chuva desenvolvido, sendo os dois condutores as placas de alumínio e o isolante o ar. [YOUNG, 2004]

Sendo assim podemos utilizar as fórmulas para capacitor para descrever os fenômenos físicos do sensor de chuva. Abaixo segue Equação 1 que define a capacitância de um capacitor.

$$C_o = \frac{Q}{V_{ab}} \quad (1)$$

onde C_o é a capacitância do capacitor, Q é a carga positiva e negativa referente a cada condutor e V_{ab} é a diferença de potencial fixa entre os condutores.

Quanto maior for à capacitância de um capacitor, maior será o módulo de Q da carga em cada condutor para uma dada diferença de potencial e, portanto, maior a energia armazenada. [YOUNG, 2004]

Quando um espaço entre dois condutores de um capacitor é ocupado por um dielétrico, a capacitância aumenta de um fator K , que é uma característica do dielétrico. Esse fenômeno foi descoberto experimentalmente por Michael Faraday.

Esse aumento se dá porque o campo elétrico entre as placas de um capacitor é enfraquecido pelo dielétrico, sendo assim, para uma determinada carga nas placas, a diferença de potencial é reduzida e a capacitância é aumentada. [TIPLER, 2006]

Considerado um capacitor carregado isolado sem um dielétrico entre suas placas. Em seguida, admita que um material dielétrico seja inserido entre as placas, preenchendo completamente o espaço entre elas. Se o campo elétrico antes de o material dielétrico inserido é E_o , após sua inserção o campo será de acordo com a Equação 2. [TIPLER, 2006]

$$E = \frac{E_o}{\kappa} \quad (2)$$

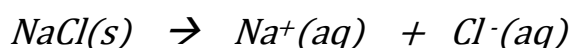
onde κ (letra grega capa) é a constante dielétrica. Com isso é possível deduzir que inserindo um dielétrico entre os condutores de um capacitor, teremos uma nova capacitância, conforme Equação 3. [TIPLER, 2006]

$$C = \kappa C_o \quad (3)$$

onde C é a capacitância com o dielétrico entre os condutores, e C_o é a capacitância sem o dielétrico.

Sendo assim, de acordo com o disposto acima, pode-se deduzir que quanto à constante dielétrica do material inserido entre as placas condutoras do capacitor, no caso o sensor de chuva, maior será a capacitância do componente. Neste projeto o material dielétrico é o ar, que possui uma constante dielétrica de aproximadamente 1,00059, isso fornece uma resistência dielétrica de 3 kV/mm. [TIPLER, 2006]

A água não é boa condutora de eletricidade, porém, a presença dos íons (partículas eletricamente positivas e negativas) em solução com a água, permite que passe corrente elétrica entre as placas de alumínio. A química abaixo mostra o que ocorre com o cloreto de sódio no estado sólido (NaCl), quando está em um meio aquoso.



está equação mostra que o cloreto de sódio teve seus íons liberados na forma do cátion Na^+ e do ânion Cl^- , quando em meio aquoso. A dissolução do sólido iônico resulta da separação dos íons de cargas opostas do material, sendo a água especialmente boa para dissolver os compostos iônicos, pois cada molécula de água tem uma extremidade positivamente carregada e outra negativamente carregada.

Com isso, quando o sensor de chuva estiver molhado, o material entre as placas condutoras será a água, que conduz eletricidade, quando o sensor estiver seco, o material entre as placas será o ar, que é um dielétrico, sendo assim só haverá condução de energia se a tensão aplicada for capaz de superar a resistência dielétrica.

A água da chuva sempre contém pelo menos um pouco de íons por conta de um fenômeno chamado acidificação. A chuva natural, não poluída, é em si um pouco ácida, devido à presença do gás carbônico natural que ao se dissolver na umidade atmosférica gera o ácido carbônico, assim: CO_2 (gás carbônico no ar) + H_2O (gotículas d'água) $\rightarrow H_2CO_3$ (ácido carbônico). [PADUA, 2004]

O ácido carbônico é um ácido fraco que se ioniza parcialmente da seguinte maneira: $\text{H}_2\text{CO}_3 \rightarrow \text{H}^+ + \text{HCO}_3^-$, portanto, a água da chuva natural possui um pH ligeiramente ácido chegando a 5,6. [PADUA, 2004]

Além disso, essa acidificação também pode ser formada na atmosfera pelo ácido sulfúrico - H_2SO_4 composto e presente naturalmente após o encontro do gás oxigênio - O_2 do ar, com o dimetil sulfeto - $\text{SO}_2 (\text{CH}_3)_2$, substância emitida por algas planctônicas. Regiões com grandes superfícies líquidas expostas disponibilizam quantidades maiores do dimetil sulfeto, (DMS). O ácido sulfúrico se destaca como sendo um partícula altamente hidrosscópica, ou seja, com grande capacidade de atrair partículas ou gotículas d'água. [PADUA, 2004]

2.3. – Microcontrolador

Os microcontroladores são componentes que em um único chip, possuem vários elementos como, uma CPU, memórias ROM e RAM, PWM, temporizadores/contadores, conversores AD, canais de comunicação e conversores analógico-digital. [MOHR, 2004]

Sistemas baseados em microprocessadores utilizam vários componentes para implementar as funções citadas anteriormente, sendo assim os microcontroladores levam vantagem na implementação de sistemas mais baratos e mais compactos, do que os baseados em microprocessadores, que por sua vez, possuem CPUs mais poderosas, conjunto de instruções menos limitadas, frequência de *clock* mais alta e costuma ter mais memória de endereçamento. [MOHR, 2004]

Disposto disto é verificado que as aplicações para estes dois tipos de sistemas são bem diferentes, os sistemas controlados por microcontrolador deve ser com menor custo e de complexidade menor, já um sistema controlado por microprocessador deve exigir uma alta capacidade de processamento.

O microcontrolador utilizado para implementação deste projeto, Figura 10, é o AT89S52, fabricado pela Atmel Corporation, este circuito integrado faz parte da família 8051 de microcontroladores, sendo assim possui o mesmo conjunto de instruções desta família.



Figura 11 - Microcontrolador

(Autor)

2.4. - A Família 8051

Os microcontroladores da família 8051 surgiram no início da década de 80, sucessor do 8048, lançadas pela INTEL, sendo bastante aceito pelo mercado. Nos dias de hoje, existem vários fabricantes com autorização para fabricação dos microcontroladores da família 8051. [NICOLOSI, 2000]

As características principais desta família são: [NICOLOSI, 2000] [MOHR, 2004]

- CPU de 8 bits;
- 64 Kbytes de endereçamento de programa (ROM externa);
- 64 Kbytes de endereçamento de memória de dados (RAM externa);
- 4 Kbytes de memória de programa (ROM interna);
- 128 bytes de memória de dados (RAM interna) e 128 bytes referentes aos registradores especiais;
- 2 *timers*/contadores de 16 bits;
- 4 portas de I/O;
- 1 interface serial;
- Processador para operação em bits (Booleano);
- 3 entradas de interrupção interna com dois níveis de prioridade;
- 2 entradas de interrupção externa com dois níveis de prioridade;
- Instrução direta de divisão e multiplicação;
- Ciclos típicos de instrução de 1 e 2 μ s a 12MHz.

A família do 8051 destaca-se das demais pelo motivo dela ser fabricada por várias empresas, e sendo assim, existe uma grande concorrência pelo mercado consumidor deste tipo de microcontrolador, o tornando mais acessível no mercado.

2.4.1. – Pinagem do 8051

Os pinos do microcontrolador têm a função de interagir com o resto do circuito, essa interação é feita por meio de sinais elétricos. A Figura 11 representa a configuração da pinagem do 8051.

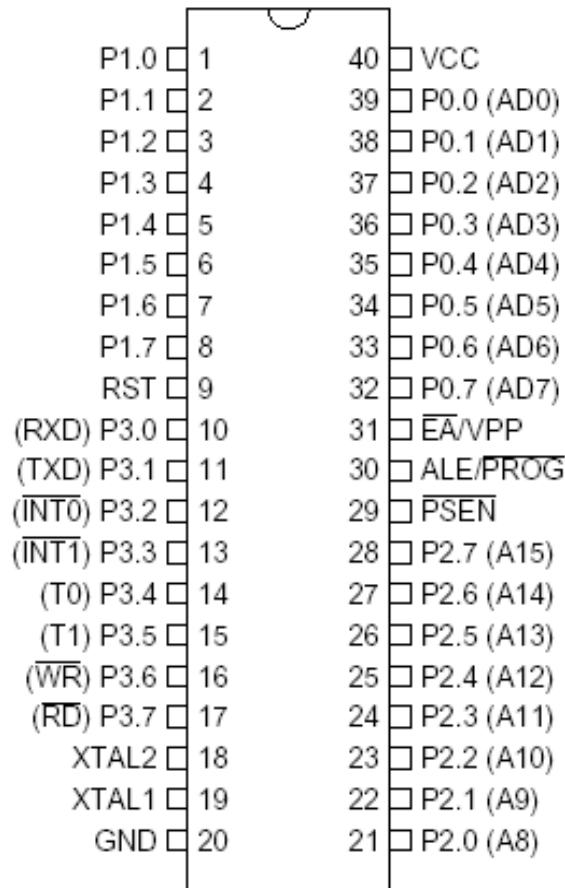


Figura 12 – Pinagem do Microcontrolador

Fonte: Datasheet do Microcontrolador AT89S52 (em anexo)

Os pinos do microcontrolador descrito têm funções diferentes entre si, abaixo são descritos o símbolo, a numeração da pinagem, o nome do pino e sua descrição e principais funções. [NICOLOSI, 2000] [MOHR, 2004]

VCC – pino 40 – Fonte de Alimentação – Entrada do dispositivo da fonte de alimentação.

VSS – pino 20 – Terra – Entrada do terra no circuito (GND).

P0.0 a P0.7 – pinos 39 a 32 – Porta 0 ou barramento de endereços menos significativos e dados multiplexados no tempo, quando se utiliza memória ROM e/ou RAM externa – A porta 0 é uma porta de entrada e saída bidirecional de 8 bits de dreno aberto. Operando como uma porta de saída, cada pino pode absorver oito entradas LS TTL. Escrevendo o 1 lógico nos pinos da porta 0, eles flutuam, e esses estados podem ser utilizados como entradas de alta impedância. A porta zero é também o barramento de endereços menos significativos multiplexados no tempo com o barramento de dados durante o acesso a uma memória de programa ou de dados externa. Nessa aplicação, utilizam-se *pull-ups* internos ao escrever o 1 lógico e pode-se fornecer ou absorver até oito entradas LS TTL. A porta 0 também recebe os bytes de códigos durante a programação da EPROM e envia os bytes de códigos durante a verificação do programa gravado na ROM e EPROM.

P1.0 a P1.7 – pinos 1 a 8 – Porta 1 é uma porta de entrada e saída bidirecional de 8 bits com *pull-ups* internos. Os buffers de saída podem fornecer ou absorver quatro entradas LS TTL. Os pinos da porta 1, que têm o 1 lógico em suas saídas, são levados a 1 lógico pelos *pull-ups* internos e, nesse caso, podem ser utilizados como entrada. Com a porta 1 funcionando como entrada, seus pinos, que são externamente levados para zero lógico, fornecerá corrente devido aos seus *pulls-ups* internos. A porta 1 também recebe o byte de endereço menos significativo

durante a programação da EPROM e durante a verificação do programa gravado da ROM e EPROM.

P2.0 a P2.7 – pinos 21 a 28 – Porta 2 ou barramento de endereços mais significativos quando se utiliza a memória ROM e/ou RAM externa – A porta 2 é uma porta de entrada e saída bidirecional de 8 bits com *pull-ups* internos. Os buffers de saída podem fornecer ou absorver quatro entradas LS TTL. Os pinos da porta 2, que têm o 1 lógico em suas saídas, são levados a 1 lógico pelos *pull-ups* internos e, nesse caso, podem ser utilizados como entrada. Com a porta 2 funcionando como entrada, seus pinos, que são externamente levados para zero lógico, fornecerá corrente devido aos seus *pull-ups* internos. A porta 2 também emite o byte do endereço mais significativo durante a programação da EPROM, durante a busca da memória de programa externa e durante o acesso à memória de dados externa que utiliza 16 bits de endereçamento. Nessa aplicação, a porta utiliza *pull-ups* internos quando emite 1 lógico. Durante o acesso à memória de dados externa que utiliza endereçamento de 8 bits, a porta 2 emite o conteúdo do registrador de função especial P2. A porta 2 também recebe o byte do endereço mais significativo durante a programação da EPROM e durante a verificação da ROM e EPROM.

P3.0 a P3.7 – pinos 10 a 17 – Porta 3 ou pinos de recepção e transmissão serial, interrupção externa 1 e 0, entrada de *clock* externa do *timer* 0 e 1, sinal de escrita e leitura de memória RAM externa – A porta 3 é uma porta de entrada e saída bidirecional de 8 bits com *pull-ups* internos. Os buffers de saída podem fornecer ou absorver quatro entradas LS TTL. Os pinos da porta 3 que têm 1 lógico em suas saídas, são levados a 1 lógico pelos *pull-ups* internos e, nesse caso,

podem ser utilizados como entrada. Com a porta 3 funcionando como entrada, seus pinos, que são externamente levados para 0 lógico, fornecerá corrente devido aos seus *pull-ups* internos.

RST – pino 9 – *Reset* – Entrada de *Reset*. Um nível lógico alto nesse pino por dois ciclos de máquina: enquanto o oscilador está sendo executado, *reseta* o dispositivo (inicializa alguns registradores internos com valores predefinidos pelo fabricante).

ALE/PROG\ – pino 30 – *Address Latch Enable/PROG* (pulso habilitador de captura de endereço) – Pulso de saída que indica a um dispositivo externo que ele deve captar o sinal de endereço no barramento de endereço e os dados que estão multiplexados no tempo. Esse pino também serve como entrada do pulso de programação durante a programação da EPROM. Em operação normal, o sinal de ALE é emitido a uma razão constante de 1/6 da frequência do oscilador e pode ser utilizado para fins de *clock* ou temporizador externo. Observe que um pulso de ALE é emitido a cada acesso à memória de dados externa.

PSEN\ – pino 29 – *Program Store Enable* (pulso habilitador de armazenamento de programa) – É o pulso de leitura para a memória de programa externa. Quando o dispositivo está executando códigos da memória de programa externa, PSEN\ é ativado duas vezes a cada ciclo de máquina, exceto quando existe acesso à memória de dados externa.

EA\ /VPP – pino 31 – *External Access Enable Programming Supply Voltage* – EA\ deve ser ligado a VSS para habilitar o dispositivo a buscar códigos da memória

de programa externa no endereço inicial de 0000h até FFFFh. EA\ deve ser ligado a VSS para a execução do programa contido na memória ROM/EPROM interna. Se o *Security Bit* (bit de segurança) na EPROM é programado, o dispositivo não buscará códigos de qualquer local de memória de programa externo. Esse pino também recebe a fonte de alimentação de programação de 21 V durante a programação da EPROM.

XTAL1 – pino 19 – Entrada do amplificador oscilador inversor.

XTAL2 – pino 18 – Saída do amplificador oscilador inversor.

2.4.2. – Interrupção

A interrupção pode ser um evento de origem interna ou externa, com isso é gerado um comando que obriga a suspensão das atividades do microcontrolador, tornando assim possível atender ao evento que solicitou a interrupção. O microcontrolador para a rotina que estava executando para atender a uma nova solicitação como uma sub-rotina que só que disparada por um evento externo. [NICOLOSI, 2000]

A ocorrência de uma interrupção não é um evento previsto, podendo ocorrer a qualquer momento, sendo assim após o fim da instrução de interrupção é necessário incluir o comando RETI (*Return Interruption*), para que quando acabem as instruções de interrupção, o programa consiga voltar onde estava antes de acontecer a interrupção. [NICOLOSI, 2000]

A interrupção pode ser de grande auxílio para eventos importantes, que podem receber a atenção imediata do processador, tornando mais fácil o desenvolvimento de sistemas processados.

2.5. – Linguagem de Programação Assembly

Uma pequena parte do algoritmo deste projeto utiliza a linguagem Assembly. A representação do Assembly é feita por mnemônicos, que são códigos alfanuméricos utilizados para tornar o entendimento dos programadores mais fáceis, sendo assim esta linguagem não é uma linguagem de máquina, como muitas vezes é confundida. Porém, uma instrução em linguagem Assembly é correspondente a uma instrução em linguagem de máquina, tornando-se uma grande vantagem sobre as linguagens de alto nível que tem seus programas mais extensos. [GIMENEZ, 2002]

Apesar da linguagem de programação Assembly ser parecida com a linguagem de máquina, até em número de instruções, ela não é uma linguagem de máquina como já foi dito. Sendo assim, após ser elaborado o programa em linguagem em Assembly, temos que compilá-lo utilizando o Assembler, que é o aplicativo responsável em transformar a linguagem Assembly em linguagem de máquina.

Linguagem de máquina é o nível mais baixo de programação, é a linguagem entendida pelo microcontrolador, os dados são representados em nível binário. Por esta razão são utilizados os mnemônicos na linguagem Assembly tornando-a tão parecida com a linguagem de máquina.

Este projeto utiliza apenas algumas linhas de comando nesta linguagem, isso se mostrou necessário para utilização de um display, que segundo fonte bibliográfica [TERROSO, 2009], utiliza alguns temporizadores que foram programados nesta linguagem.

2.6. – Linguagem de Programação C

A maior parte do algoritmo utilizado neste projeto é escrito em linguagem C, Essa linguagem de programação é uma linguagem de alto nível, que apesar de ter sido criada na década de 70, ainda é largamente utilizada por programadores. [GIMENEZ, 2002]

Este programa é composto por funções, isto é, porções menores de código que realizam determinadas tarefas, e por dados, ou seja, variáveis ou tabelas que são inicializadas antes do início do programa. Existe uma função especial denominada *main*, onde a execução do programa se inicia.

Em linguagem C, a organização do código em arquivos tem um significado semântico. O arquivo fonte principal, passado como argumento para o compilador, define uma unidade de compilação.

Uma unidade pode importar definições usando a diretiva *#include* ou apenas declarando algum identificador como externo.

Esta linguagem permite um modelo de compilação separado, isto é, você pode dividir o programa em várias unidades independentes que são compiladas separadamente e depois são concatenadas com o link editor para construir o

programa final. Normalmente, cada módulo é escrito num arquivo texto separado que contém funções ou declarações de dados. As interfaces entre os módulos são escritas em "*header files*" (arquivos de cabeçalho) que descrevem tipos ou funções visíveis para vários módulos do programa. Estes arquivos possuem a extensão ".h".

Toda função possui uma lista de parâmetros, um corpo e, eventualmente, um valor de retorno. O corpo pode conter declarações de variáveis locais, ou seja, variáveis que são ativadas quando a execução alcançar o corpo da função.

2.7. – Circuitos Integrados e Componentes

Neste tópico, é descrito algumas informações técnicas dos componentes utilizados na implementação do sistema.

2.7.1. – Display

Este projeto usa o Display de Cristal Líquido com dimensões 16x2, e numeração MGD1602B. O Display é utilizado para interação com o usuário do sistema, nele é escrito as ações efetuadas pelo sistema. Ele possui para escrita, duas colunas com possibilidade de utilização de dezesseis caracteres em cada uma delas. A Figura 12 é uma foto do display mencionado.



Figura 13 - Display

(Autor)

Este Display possui dezesseis pinos dispostos conforme Quadro 1, sendo que dois deles são para o funcionamento do *Black Light* que é o acendimento com luz esverdeado do fundo do display, porém nem todos os displays os possuem.

Nº do Pino	Símbolo	Função
1	Vss	Aterramento
2	Vdd	+5volts
3	Vo	Ajuste de Contraste
4	RS	Seleção de Registro
5	R/W	Leitura/Escrita
6	E	Permissão de sinal
7	DB0	Trafego de Dados
8	DB1	Trafego de Dados
9	DB2	Trafego de Dados
10	DB3	Trafego de Dados
11	DB4	Trafego de Dados
12	DB5	Trafego de Dados
13	DB6	Trafego de Dados
14	DB7	Trafego de Dados
15	A	+5volts para <i>Black Light</i>
16	K	Aterramento para <i>Black Light</i>

Quadro 1 – Pinagem do Display

O Display possui alguns comandos previamente definidos, para que ele reconheça e interprete os dados enviados por um microcontrolador, o Quadro 2 mostra alguns comandos utilizados para escrita no Display, através do microcontrolador.

Comando	Descrição do comando
38 H	Informa ao display que ele possui 2 linhas e 16 colunas e cada cursor é composto por uma resolução de 5X8 pixels
C0 H	Avisa ao display para ele começar a escrever na linha 2, coluna 1
80 H	Avisa ao display para ele começar a escrever na linha 1, coluna 1
0E H	Aciona cursor
06 H	Habilita escrita da esquerda para direita
01 H	Limpa o display

Quadro 2 – Comandos do Display

2.6.2. – L298N

O L298 é um circuito monolítico integrado de 15 ligações, ele é um excitador de tensão com duas Ponte-H, já comentada anteriormente, que são projetadas para aceitar padrões lógicos TTL (*Transistor-Transistor Logic*), tornando possível controlar cargas indutivas. A Figura 13 mostra a pinagem deste CI.

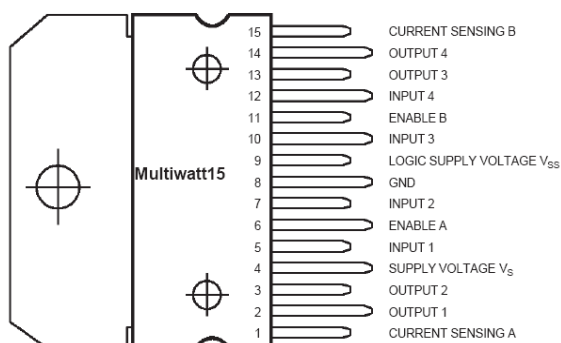


Figura 14 – Pinagem do CI L298N

Fonte: Datasheet do CI L298N (em anexo)

Sendo assim, além desse circuito ter duas Ponte-H, ele recebe os dados direto do microcontrolador em formato TTL, e enviar ao motor de passo a tensão desejada para o seu funcionamento.

O CI L298N pode operar com tensão de saída de até 46 volts, tornando muito útil para controle de motor de passo bipolar, que é o caso deste projeto, ele suporta corrente de até 4 ampère e tem baixa tensão de saturação, este CI também possui proteção contra temperatura excedente, e o valor lógico “0” dele é ativo com tensão de entrada de até 1,5 volts, tornando-se bastante eficiente contra ruídos.

A Figura 14 mostra o diagrama de blocos do CI L298, onde é possível verificar as duas Ponte-H.

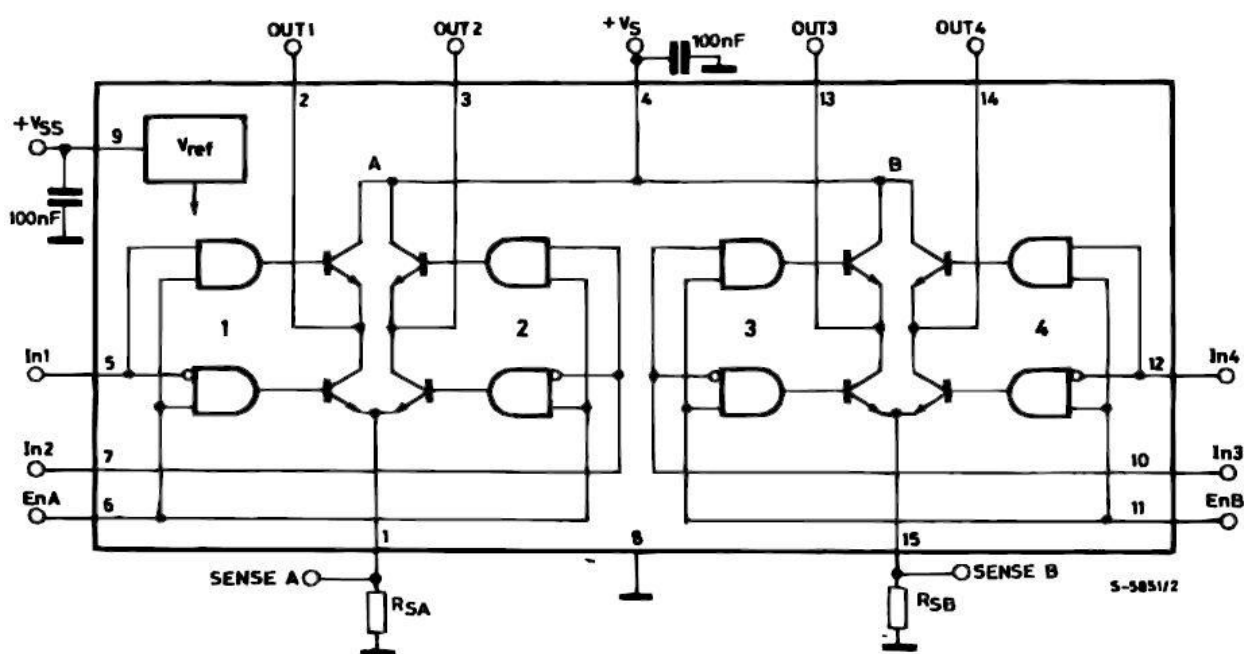


Figura 15 – Diagrama de Blocos L298

Fonte: Datasheet do CI L298N (em anexo)

CAPÍTULO 3 – SISTEMA DE AUTOMAÇÃO DE JANELAS

3.1. – Apresentação Geral

Este projeto baseia-se na intenção de trazer comodidade e segurança para uma residência, na forma de automatização das janelas.

O protótipo apresentado é composto por um sensor de chuva, um sensor de janela fechada, um sensor de janela aberta, um botão para abrir a janela, um botão para fechar a janela, uma chave liga/desliga que pode colocar em modo manual ou automático, um display de LCD, um circuito integrado L298N e um microcontrolador da família 8051.

O protótipo tem a função de abrir e fechar a janela de acordo com os critérios especificados. Primeiramente, o sistema nunca deve deixar a janela aberta enquanto estiver chovendo, pois este é o propósito da construção dele, porém, o sistema tem várias outras formas de controle da janela. Conforme já foi dito, o sistema conta com uma chave liga/desliga que controla o modo de atuação do sistema, se é automático ou manual, sendo assim, os parâmetros de entrada deste sistema são: sensor de chuva (mostra se está chovendo ou não); chave manual/automático (controla o modo de atuação do sistema); botão de abre e botão de fecha janela.

De acordo com os parâmetros de entrada no sistema, é possível, através do microcontrolador, fazer inúmeras combinações onde cada combinação pode abrir ou fechar a janela, bem como deixar a janela aberta ou fechada. No Quadro 3 são mostras todas as combinações, bem como as ações correspondentes a cada uma delas.

Modo de Operação	Condição Climática	Acionamento de Botão	Ação do motor
Modo Automático	Com Chuva	Bot 5	Fecha janela
		Bot 6	
	Sem Chuva	Bot 5	Abre Janela
		Bot 6	
Modo Manual	Com Chuva	Bot 5	Fecha janela
		Bot 6	
	Sem Chuva	Bot 5	Abre Janela
		Bot 6	Fecha Janela

Quadro 3 – Possibilidades do sistema

Além de abrir e fechar a janela conforme especificado, o sistema conta também com um display onde todas as situações são escritas. No display sempre está sendo mostrado se o tempo está limpo ou se está chovendo, nele também será indicado se o sistema se encontra no modo manual ou automático. Caso a janela esteja sendo fechada ou aberta, o display mostra em sua tela a ação ocorrendo (exemplo: “FECHANDO JANELA”), quando ela acabar o display mostra a situação final (exemplo: “JANELA FECHADA”).

3.1.1. – Fluxograma Geral

Este tópico descreve o fluxograma do projeto, nele é possível verificar todas as tomadas de decisões que o sistema toma durante sua execução.

Na Figura 15 pode-se observar o fluxograma do projeto. É importante observar a concordância da Figura 15 com a Tabela 3. Através dele que foi desenvolvido a programação inserida no microcontrolador.

Neste fluxograma, quando o botão 1 for acionado “S”, considere mono manual, quando não acionado “N” considere modo automático.

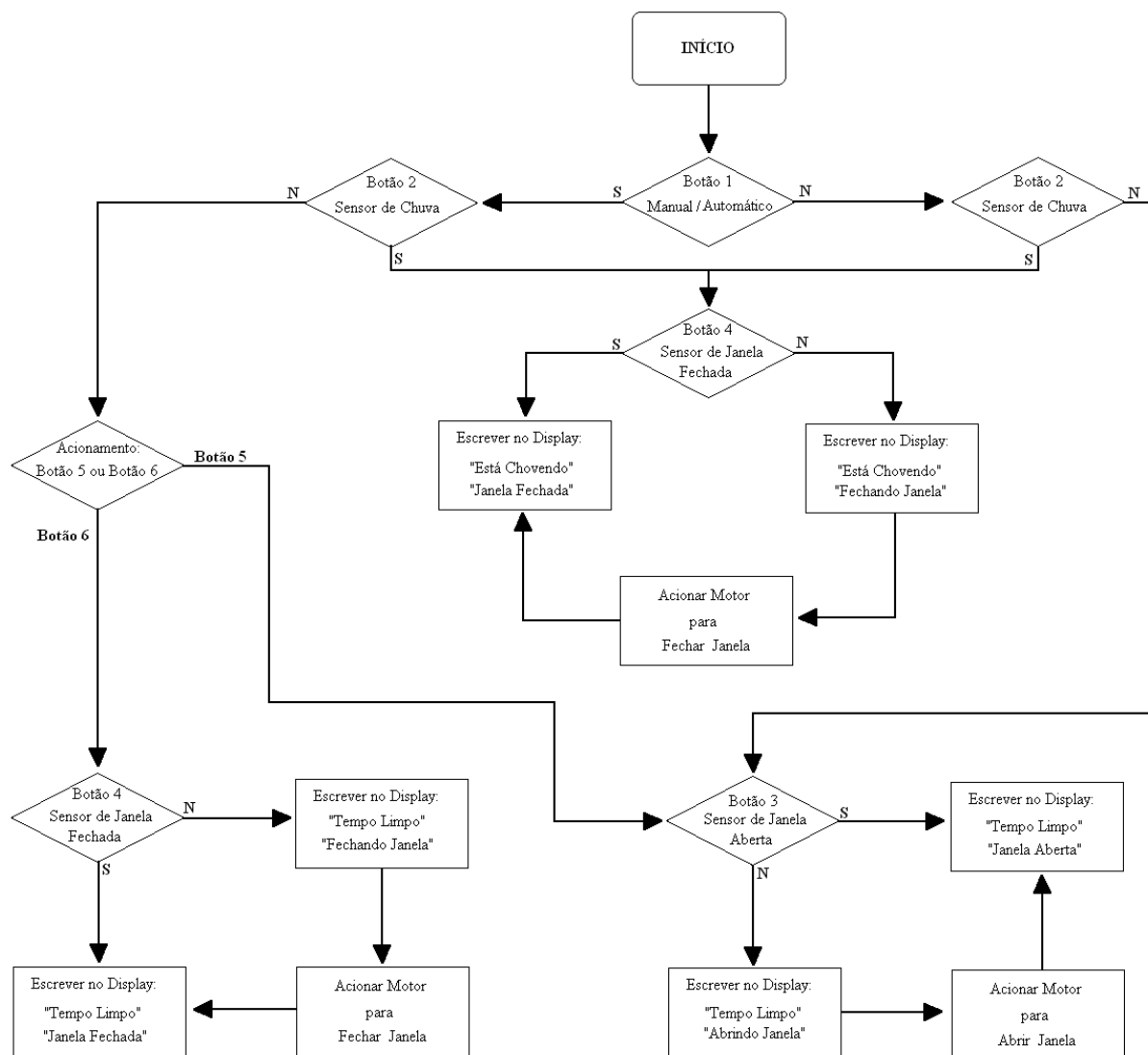


Figura 16 – Fluxograma do sistema

(Autor)

O termo Fluxograma designa uma representação gráfica de um determinado processo ou fluxo de trabalho, efetuado geralmente com recurso das figuras geométricas normalizadas e a setas unindo-as. Através desta representação gráfica é possível compreender de forma rápida e fácil a transição de informações ou documentos entre os elementos que participam no processo em causa. [NUNES, 2009]

3.1.2. - Visão geral do Projeto

Na Figura 16 são mostras de forma simplificada as ligações realizadas entre os elementos do sistema.

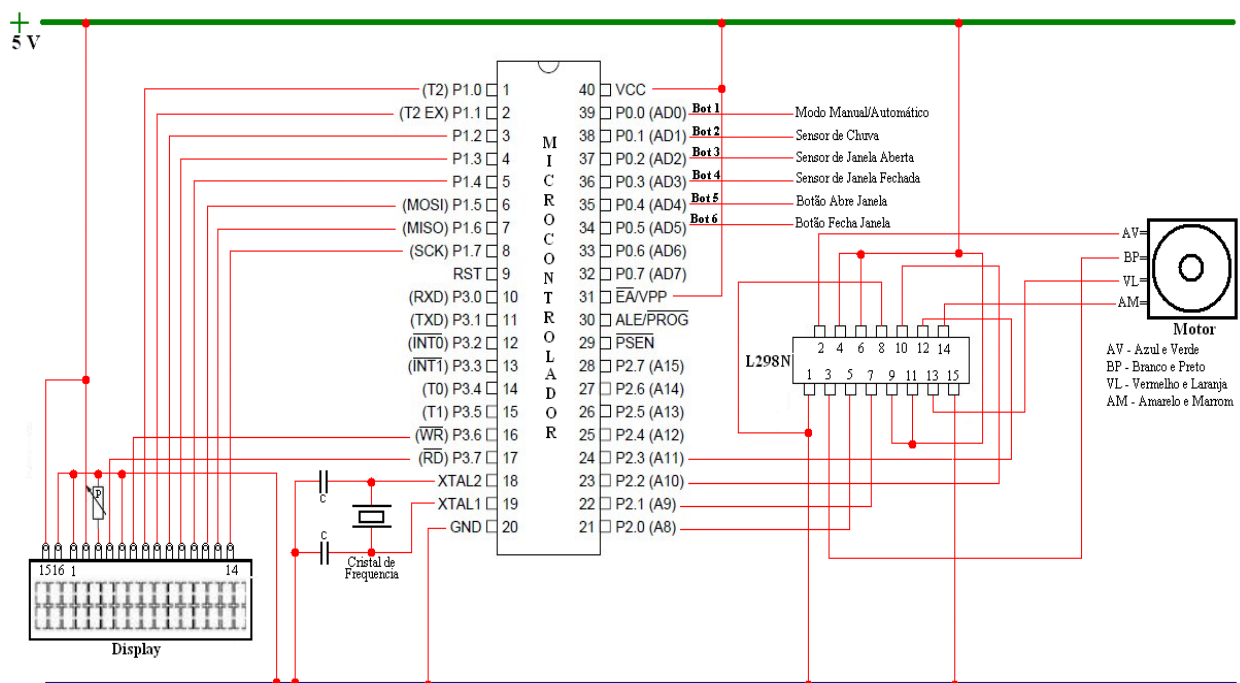


Figura 17 – Visão Geral do Projeto

(Autor)

O microcontrolador envia as instruções de acordo com os sinais de entrada que estão ligadas à porta P0 do microcontrolador, as instruções de escrita dada ao display, saem da porta P1 do microcontrolador, e as instruções para mover o motor de passo saem da porta P2 do microcontrolador.

O circuito integrado L298N faz a ponte entre o motor de passo e o microcontrolador tem a função de elevar a tensão aplicada ao motor de passo, para que aí sim ele tenha força para girar.

O motor de passo é engrenado à janela para que ela possa abrir e fechar, apesar da janela não estar representada na Figura 16.

3.2. – Controle do Motor de Passo

O motor de passo utilizado neste projeto é um motor de passo híbrido e bipolar, tendo assim um ganho de torque comparado a outras especificações, da marca *Tamagawa* modelo *TS 3218 N814*.

Os motores de passo trabalham através de sinais digitais (0 e 1), onde “0” é ausência de energia e “1” envio de sinal elétrico. Sendo assim, é possível estabelecer uma lógica de funcionamento, de tal forma que, as bobinas sejam energizadas em uma ordem que gere os passos no motor colocando-o em movimento.

No Quadro 4 são mostras as bobinas que devem estar ligadas, e as bobinas que devem estar desligadas a cada número de passo, para que o motor de os passos e conseqüentemente gire.

Nº do Passo	B0	B1	B2	B3
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Quadro 4 – Passos do motor

Abaixo segue fragmento da programação, com a função de girar o motor de passo, é importante frisar que para que o motor gire para o outro lado a seqüência de bobinas energizadas devem ser invertida.

```
(...)
while(bot4==0)
{
    for(i=1;i<5;i++)
    {
        tempo();
        switch(i)
        {
            case 1:
                led1=1;
                led2=0;
                led3=0;
                led4=0;
                break;
            case 2:
                led1=0;
                led2=1;
                led3=0;
                led4=0;
                break;
            case 3:
                led1=0;
                led2=0;
                led3=1;
                led4=0;
                break;
            case 4:
                led1=0;
                led2=0;
                led3=0;
                led4=1;
                break;
            default:
                break;
        }
    }
}
```

Neste programa, “bot4” representa o sensor de janela fechada, “led 1, 2, 3 e 4” representa as saídas do microcontrolador que enviam sinais ao motor. Assim sendo, este fragmento do programa tem a função de movimentar o motor, fechando assim a janela, até que o sensor de janela fechada seja ativado.

O microcontrolador é capaz de mandar o sinal em ordem certa para o motor de passo, porém, a tensão enviada por ele não é capaz de energizar o suficiente as bobinas do motor, portanto, o circuito integrado L298N faz a ligação entre o microcontrolador e o motor de passo, recebendo o sinal com a tensão baixa e enviando um sinal com a tensão apropriada.

3.3. – Controle do sensor de chuva

O sensor de chuva tem um de seus fios conectado na porta P0.1 do microcontrolador e o outro fio conectado no pólo positivo da fonte de alimentação. Sua função é enviar corrente elétrica ao microcontrolador quando estiver chovendo e interromper a corrente elétrica na ausência de chuva, para que assim, o microcontrolador tome as devidas providências.

A referência do sensor de chuva na programação desenvolvida é “bot2”, como a intenção deste projeto é sempre fechar a janela enquanto houver chuva, o parâmetro “bot2” quando ativo, sempre chama a função para fechar a janela. Abaixo segue um fragmento do programa principal onde “bot1” é referente à chave de mudança de modo manual e automático. Sendo assim este fragmento mostra um

loop onde enquanto estiver chovendo a janela será ou permanecerá fechada, ação executada pela função “chuva_fecha_janela()”.

```
(...)  
x2=100;  
while(bot1==0 && bot2==1)  
{  
    if(x2==100)  
    {  
        chuva_fecha_janela();  
        x2=0;  
    }  
}  
(...)
```

3.4. – Controle do Display de LCD

O controle do display de LCD foi desenvolvido a partir de um exercício proposto, [TERROSO, 2009], as configurações do display também tiveram como referência este exercício.

Abaixo, segue função que configura o display para que o microcontrolador reconheça os dados enviados.

```
void config_lcd(void)  
{  
    dado = 0x38;  
    wr_cmd();  
    dado = 0x07;  
    wr_cmd();  
    dado = 0x0E;  
    wr_cmd();  
    dado = 0x06;  
    wr_cmd();  
    dado = 0x01;  
    wr_cmd();  
}
```

Para utilização das funções de escrita na tela do display é utilizado duas funções marcadoras de tempo que foram escritas em assembly. Este trecho também

foi retirado do exercício já citado. Abaixo segue as duas funções marcadoras de tempo:

```
void delay_6us(void)          //rotina de delay de 2 microsegundos
{
    _asm                      //inicio de trecho assembly
    0$: mov  r0, #6
    1$: djnz r0, 1$
    _endasm;                  //fim do trecho assembly

    // 1 ciclo com 12Mhz = 1us => 1 ciclo com 27Mhz = 0.444444444444 microsegundos
    // #ciclos = 1c+(2c)*6
    // #ciclos = 13
    // #tempo = 13 * 0.444444444444 = 5.777 microsegundos
}

//*****

void delay_6ms(void)          //rotina de delay de 2 milisegundos
{
    _asm                      //inicio de trecho assembly
    1$: mov  r1, #27           //1c
    2$: mov  r2, #255         //1c
    3$: djnz r2, 3$           //2c
        djnz r1, 2$           //2c
    _endasm;                  //fim do trecho assembly

    // 1 ciclo com 12Mhz = 1us => 1 ciclo com 27Mhz = 0.444444444444 microsegundos
    // #ciclos = 1c+[1c+(2c)*255+2c]*27
    // #ciclos = 13852
    // tempo = 13852 * 0.444444444444 = 6.156 milisegundos
}
```

3.5. – Controle do Microcontrolador

O microcontrolador possui o controle de todas as ações ocorrentes no sistema, mas para isso, é necessário desenvolver uma programação capaz de identificar e tratar todas as informações do sistema.

3.5.1. – Programação de controle

A programação utilizada no microcontrolador é predominantemente em C, apenas uma pequena parte já citada foi desenvolvida em assembly. Para que a

programação em C desenvolvida reconheça que está trabalhando em um microcontrolador da família 8051 faz-se necessário definir e nomear as portas do microcontrolador que serão utilizadas. Abaixo segue trecho do código onde são feitas essas definições.

```
#include "8051.h"

#define dado P1 //declarando porta P1 para envio de dados ao LCD
#define rs P3_7 //declarando os sinais P3.7 ao sinal RS
#define en P3_6 //declarando os sinais P3.6 ao sinal EN
#define led1 P2_0 // declara porta para controlar passo 1 do motor
#define led2 P2_1 // declara porta para controlar passo 2 do motor
#define led3 P2_2 // declara porta para controlar passo 3 do motor
#define led4 P2_3 // declara porta para controlar passo 4 do motor
#define bot1 P0_0 // modo manual / automatico
#define bot2 P0_1 // sensor de chuva
#define bot3 P0_2 // sensor de janela aberta
#define bot4 P0_3 // sensor de janela fechada
#define bot5 P0_4 // botão abre janela
#define bot6 P0_5 // botão fecha janela
```

Este software contém várias funções, que é um pedaço de código que faz alguma tarefa específica e pode ser chamado de qualquer parte do programa quantas vezes desejarmos, as funções utilizadas estão abaixo.

```
void config_lcd(void); //função configura lcd
void linha1(void); //função escreve linha 1 do lcd
void linha2(void); //função escreve linha 2 do lcd
void escreve_char(char msn[]); //função escreve string
void delay_6us(void); //função de delay de 6 microsegundos
void delay_6ms(void); //função de delay de 8 milisegundos
void wr_cmd(void); //função para envio de configuração ao lcd
void wr_char(void); //função para envio de string ao lcd
void tempo(void); //função de intervalo de tempo entre passos do motor
void tempo_limpo_abre_janela(void); //função para abrir janela enquanto o tempo estiver limpo
void chuva_fecha_janela(void); //função para fechar janela enquanto estiver chovendo
void tempo_limpo_fecha_janela(void); //função para fechar janela enquanto o tempo estiver limpo
void status_lcd(void); //função para escrever no lcd o status da janela
```


3.5.2. – Inserção do algoritmo no Microcontrolador

Para copiar o programa desenvolvido em C para o microcontrolador, é necessário usar um compilador capaz gerar um arquivo com a extensão “.hex”, que é o tipo de arquivo interpretado pelo microcontrolador.

Este projeto teve como interface para desenvolvimento e compilação do algoritmo o programa ConTEXT v0.97.6, programa este distribuído sem custos, conforme se pode observar na Figura 17.



**Figura 18 – Programa ConTEXT
(Autor)**

“Este programa, após ser desenvolvido o algoritmo em C, compila o algoritmo gerando vários tipos de arquivos, conforme Figura 18, porém com dito anteriormente o microcontrolador trabalha com o arquivo de extensão “.hex”, esta extensão é chamada de hexadecimal, pois este tipo de arquivo trabalha com números hexadecimais.















 meu_programa	40 KB	Arquivo ASM
 meu_programa	2 KB	PowerISO File
 meu_programa	12 KB	Arquivo C
 meu_programa	5 KB	Arquivo HEX
 meu_programa	1 KB	Atalho
 meu_programa	93 KB	Arquivo LST
 meu_programa	7 KB	Arquivo MAP
 meu_programa	37 KB	Arquivo SYM
 meu_programa.asm.org	40 KB	Arquivo ORG
 meu_programa.c.org	12 KB	Arquivo ORG
 meu_programa.ihx	6 KB	Arquivo IHX
 meu_programa.mem	1 KB	Arquivo MEM
 meu_programa.rel	17 KB	Arquivo REL
 meu_programa.rst	93 KB	Arquivo RST

Figura 19 – Extensões geradas pelo ConTEXT

(Autor)

Após a compilação do algoritmo e geração da extensão hexadecimal, foi utilizado neste projeto um circuito adquirido na internet capaz de fazer a gravação do arquivo no microcontrolador. O kit adquirido para gravação do arquivo foi adquirido pelo site <http://www.microgenius.com.br>, a Figura 19 mostra uma foto deste circuito.

Diversas tentativas de construir um gravador foram feitas, porém nenhum circuito construído respondeu às expectativas, sendo assim o custo do projeto teve um aumento considerável, porém, é necessário um gravador de confiança para que todos os testes possam ser feitos.

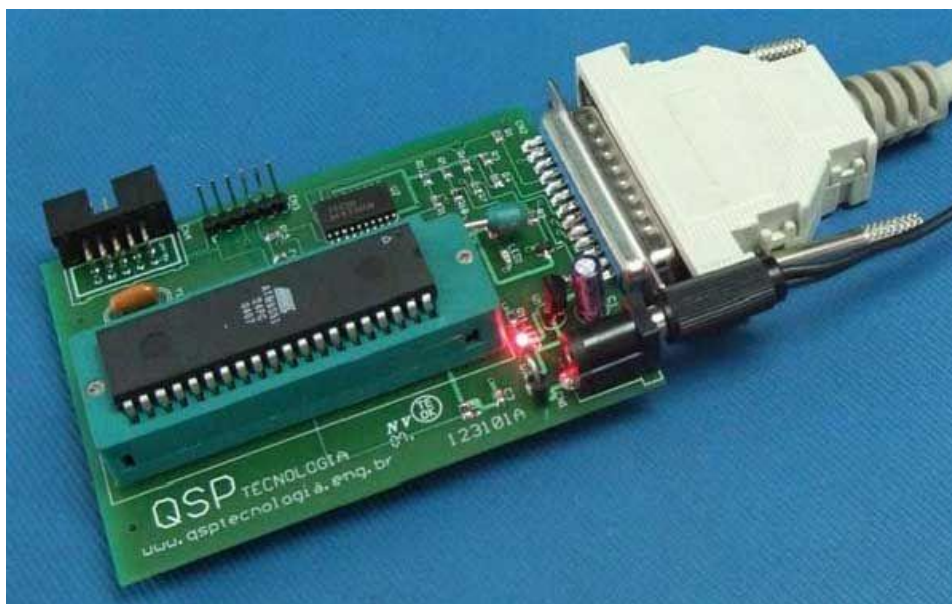


Figura 20 - Gravador de Microcontroladores AT89s51/s52/s53/8252/8253 ATMEL

[fonte: <http://www.microgenius.com.br>, novembro de 2009]

Após a gravação do arquivo hexadecimal no microcontrolador, ele está pronto para ser utilizado

3.6. – Apresentação final do circuito

Para testar o circuito, foi utilizado o *software Proteus 7*. Esta potente ferramenta permite simular diversos tipos de circuitos eletrônicos através do computador, com isso ganha-se tempo e dinheiro, uma vez que os circuitos neste *software* são de fácil montagem, e não tem risco de queimar algum componente caso algo esteja ligado errado. Este *software* também permite a utilização de microcontroladores, sendo assim após a compilação e algoritmo e geração do arquivo hexadecimal, é possível testar o circuito por completo. A Figura 20, mostra as ligações feita entre os elementos do projeto, utilizando o *software Proteus 7*.

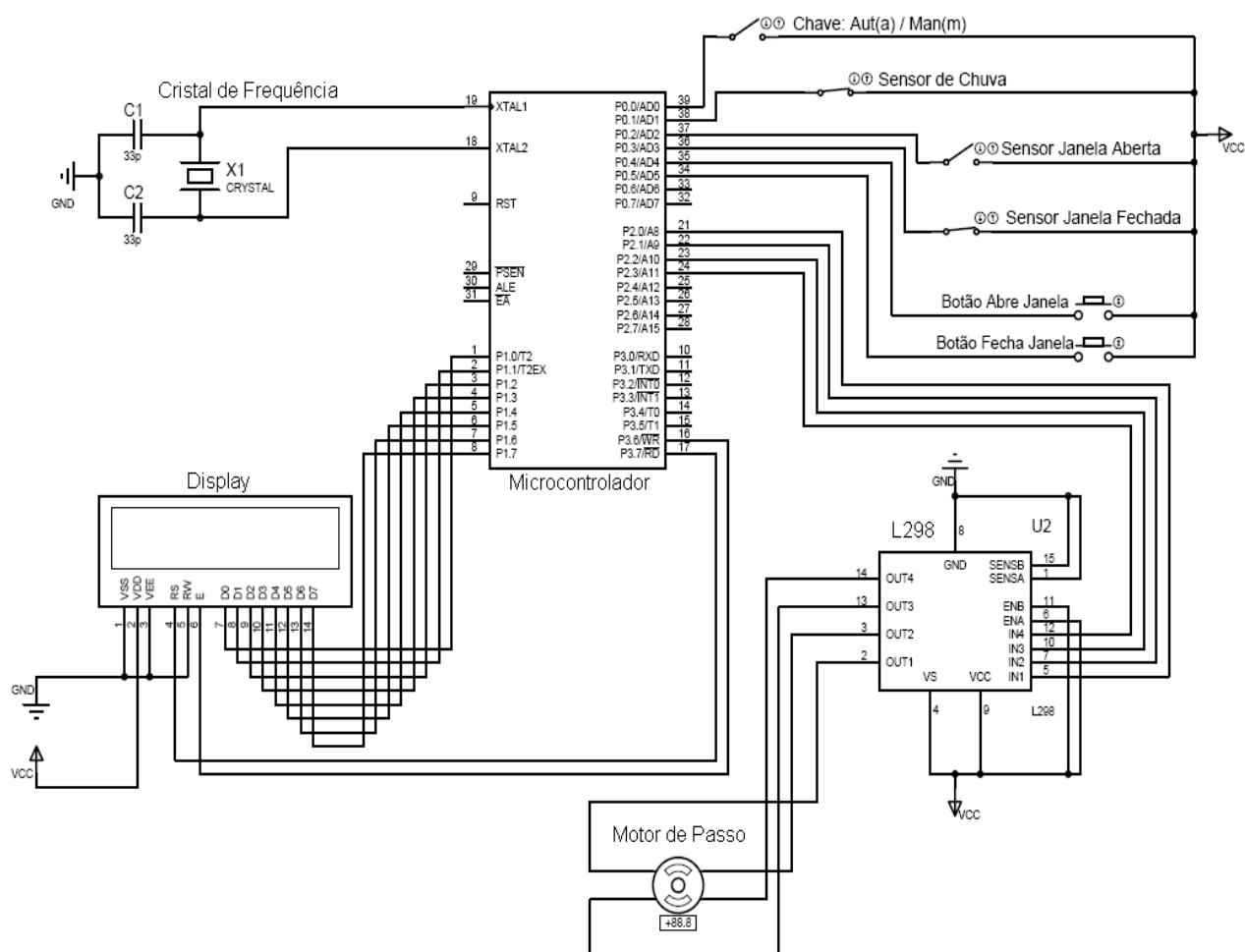


Figura 21 – Ligações do Projeto

(Autor)

O microcontrolador está ligado a todos os elementos do circuito, direta ou indiretamente, segue descrição das ligações entre o microcontrolador e os sensores e sinais de entrada, bem como a ligação com o cristal de frequência.

- Pino 18 e 19 (XTAL 2 e 1) – saída e entrada do cristal de frequência no microcontrolador.
- Pino 39 (P0.0) – Chave para controle do modo manual e automático.
- Pino 38 (P0.1) – Sensor de chuva
- Pino 37 (P0.2) – Sensor de janela aberta

- Pino 36 (P0.3) – Sensor de janela fechada
- Pino 35 (P0.4) – Botão para abrir janela.
- Pino 36 (P0.5) – Botão para fechar janela.

As ligações feitas entre o display de cristal líquido com o microcontrolador são as seguintes:

- Pino 1 – Terra.
- Pino 2 – Fonte de alimentação de 5 volts.
- Pino 3 – Potenciômetro em série com o terra
- Pino 4 – Pino 17 (P3.7) do microcontrolador
- Pino 5 – Terra.
- Pino 6 – Pino 16 (P3.6) do microcontrolador
- Pino 7 – Pino 1 (P1.0) do microcontrolador.
- Pino 8 – Pino 2 (P1.1) do microcontrolador.
- Pino 9 – Pino 3 (P1.2) do microcontrolador.
- Pino 10 – Pino 4 (P1.3) do microcontrolador.
- Pino 11 – Pino 5 (P1.4) do microcontrolador.
- Pino 12 – Pino 6 (P1.5) do microcontrolador.
- Pino 13 – Pino 7 (P1.6) do microcontrolador.
- Pino 14 – Pino 8 (P1.7) do microcontrolador.
- Pino 15 – Fonte de alimentação de 5 volts (*Black Ligth*).
- Pino 16 – Terra (*Black Light*).

Conforme já foi detalhado anteriormente, o circuito integrado L298N, faz a ligação entre o microcontrolador e o motor de passo. As portas do circuito integrado são conectadas ao motor de passo e ao microcontrolador da seguinte forma:

- Pino 1 – Terra.
- Pino 2 – Fio azul do motor de passo.
- Pino 3 – Fio branco do motor de passo.
- Pino 4 – Fonte de alimentação de 5 volts.
- Pino 5 – Pino 21 (P2.0) do microcontrolador.
- Pino 6 – Fonte de alimentação de 5 volts.
- Pino 7 – Pino 22 (P2.1) do microcontrolador.
- Pino 8 – Terra.
- Pino 9 – Fonte de alimentação de 5 volts.
- Pino 10 – Pino 23 (P2.2) do microcontrolador.
- Pino 11 – Fonte de alimentação de 5 volts.
- Pino 12 – Pino 24 (P2.3) do microcontrolador.
- Pino 13 – Fio vermelho do motor de passo.
- Pino 14 – Fio amarelo do motor de passo.
- Pino 15 – Terra.

Depois de feita essas ligações e inserido o arquivo correto no microcontrolador, o circuito está pronto. A seguir, algumas figuras mostrando o funcionamento do circuito com uma breve descrição:

A Figura 21, mostra a situação em que o sistema está operando em modo automático, que pode ser observado no display o caractere “(a)”, sendo assim como o sensor de chuva não está ativo, ou seja não está chovendo, a janela se abrirá até que o sensor de janela aberta seja ativado. Nesta situação o display mostra na primeira linha “TEMPO LIMPO (a)”, que significa que não está chovendo e que o sistema está operando no modo automático, na segundo linha do display está à informação “ABRINDO JANELA”, esta condição ocorre enquanto o sensor de janela aberta não está ativo, senso assim o motor de passo girará até que a janela seja completamente aberta.

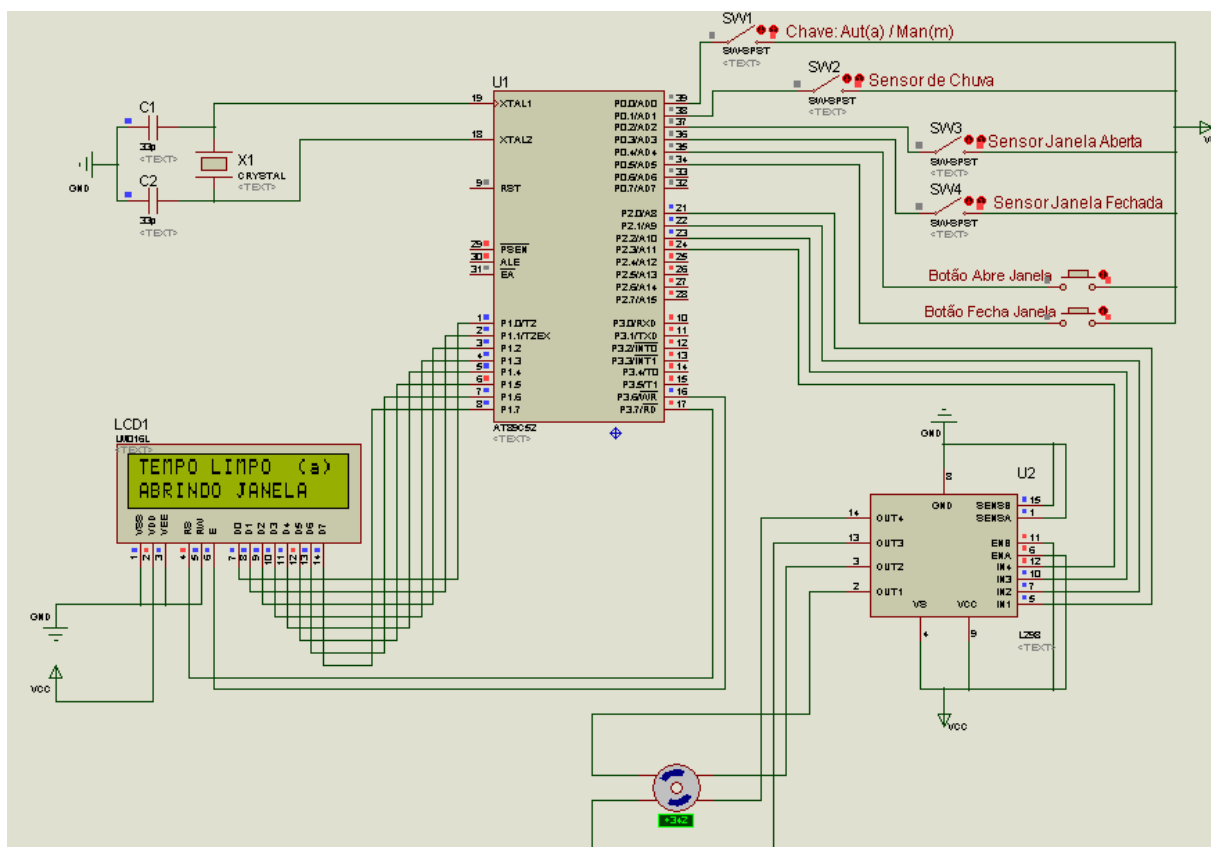


Figura 22 – LCD - Tempo Limpo (a); Abrindo Janela

(Autor)

A Figura 22, mostra a continuação da situação descrita na Figura 21, neste caso o sensor de janela aberta foi ativando, logo a janela encontra-se fechada, assim sendo o motor de passo para de funcionar e o sistema fica na espera de outra função. Nesta situação o display mostra “TEMPO LIMPO (a)” na primeira linha e “JANELA ABERTA” na segunda linha.

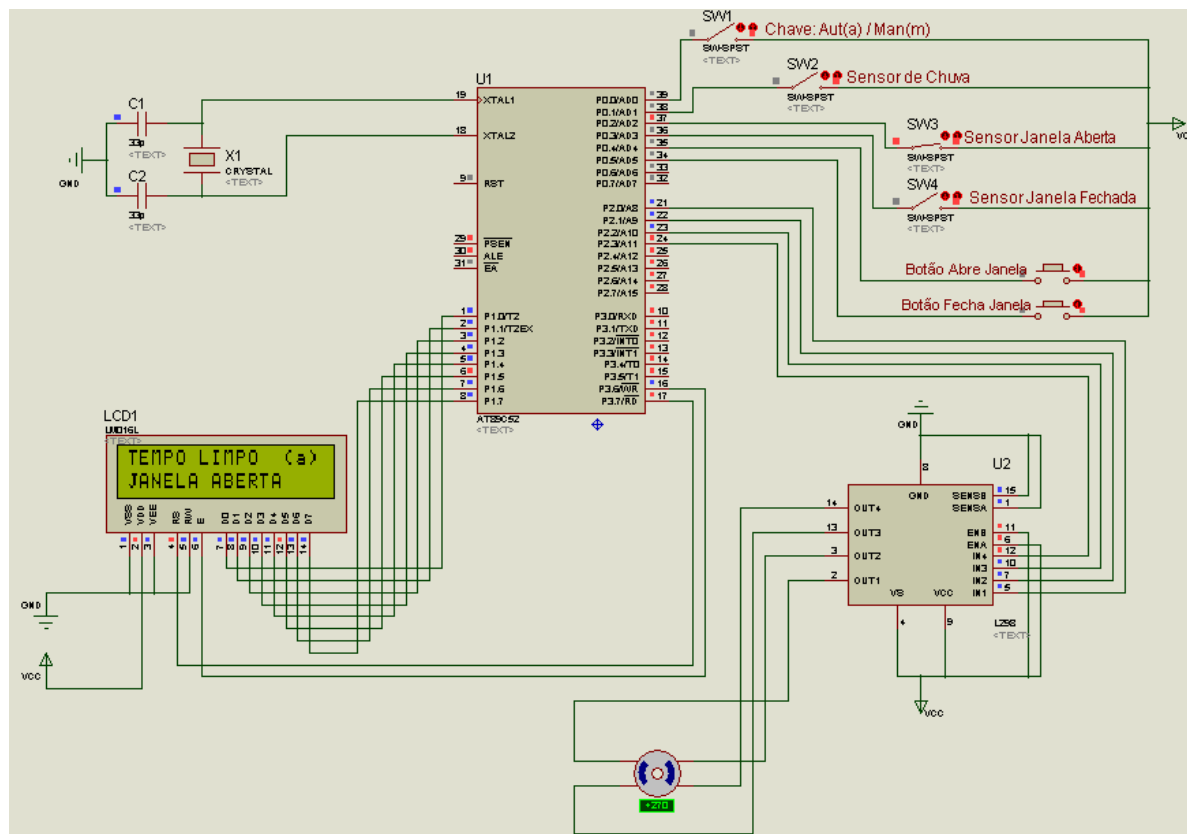


Figura 23 – LCD - Tempo Limpo (a); Janela Aberta

(Autor)

A Figura 23, mostra a situação inversa à mostrada na Figura 21, neste caso o sistema continua operando em modo automático, porém o sensor de chuva foi ativado, com isso é enviado um sinal para que o motor gire, fechando a janela, até que o sensor de janela fechada seja ativado. Nesta situação o display mostra na primeira linha “ESTA CHOVENDO (a)”, que significa que está chovendo e que o

sistema está operando no modo automático, na segunda linha do display está à informação “FECHANDO JANELA”, mostrando que o sensor de janela fechada ainda não está ativo.

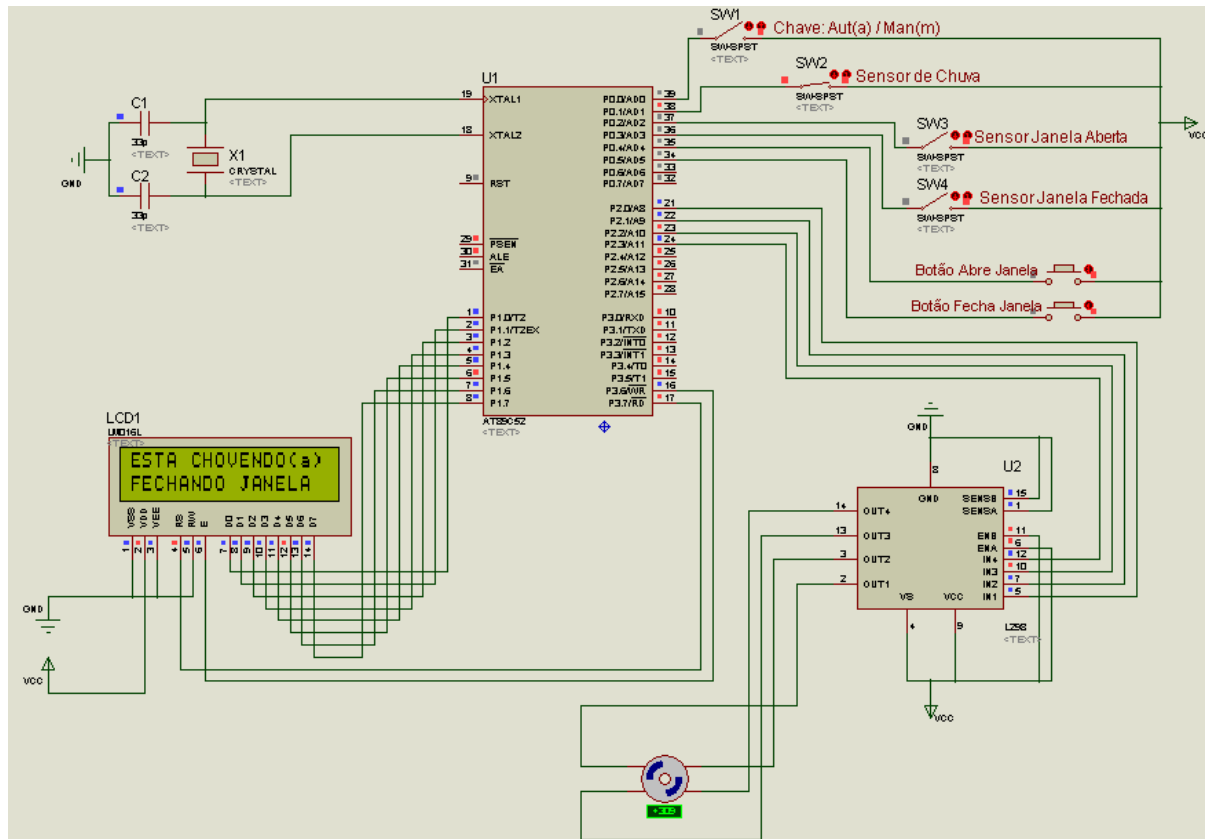


Figura 24 – LCD - Esta Chovendo (a); Fechando Janela

(Autor)

Quando o sensor de janela fechada for ativo significa que a janela está completamente fechada, assim sendo, o sistema cessa a atividade do motor e fica aguardando novas instruções, conforme Figura 24. Nesta situação o display mostra “ESTA CHOVENDO (a)” na primeira linha e “JANELA FECHADA” na segunda linha.

É importante salientar que enquanto o sistema estiver operando no modo automático, os botões de abrir e fechar a janela não tem funcionamento.

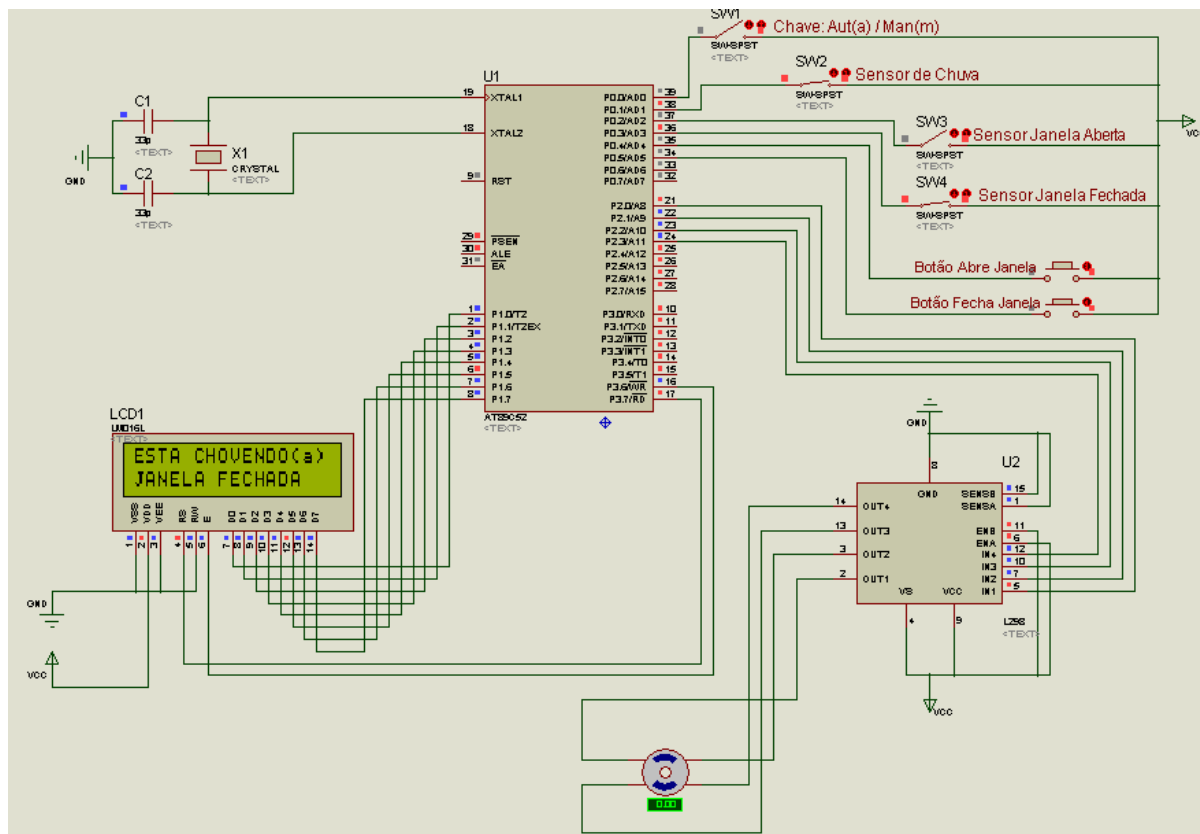


Figura 25 – LCD - Esta Chovendo (a) Janela Fechada

(Autor)

As próximas descrições de situação são feitas quando o sistema estiver operando no modo manual. Neste modo o usuário tem um controle sobre a janela, podendo abrir e fechar a janela utilizando dois botões, um para abrir e outro para fechar a janela. Porém, como a função principal do projeto é não permitir que a água da chuva entre pela janela, os botões citados só terão funcionamento enquanto não estiver chovendo, ou seja, caso não esteja chovendo e o modo de operação do sistema estiver no manual, o usuário pode abrir ou fechar a janela de acordo com sua opção.

A Figura 25 mostra uma situação em que o sistema está no modo manual e não está chovendo. Esta configuração é típica quando você acaba de mudar a chave do modo automático para o manual.

Nesta situação o display mostra “TEMPO LIMPO (m)” na primeira linha, isto informa que não está chovendo e o sistema está operando no modo manual, na segunda linha o display mostra “JANELA ABERTA”, ou seja, o sensor de janela aberta está ativado, conforme pode ser verificado na Figura 25.

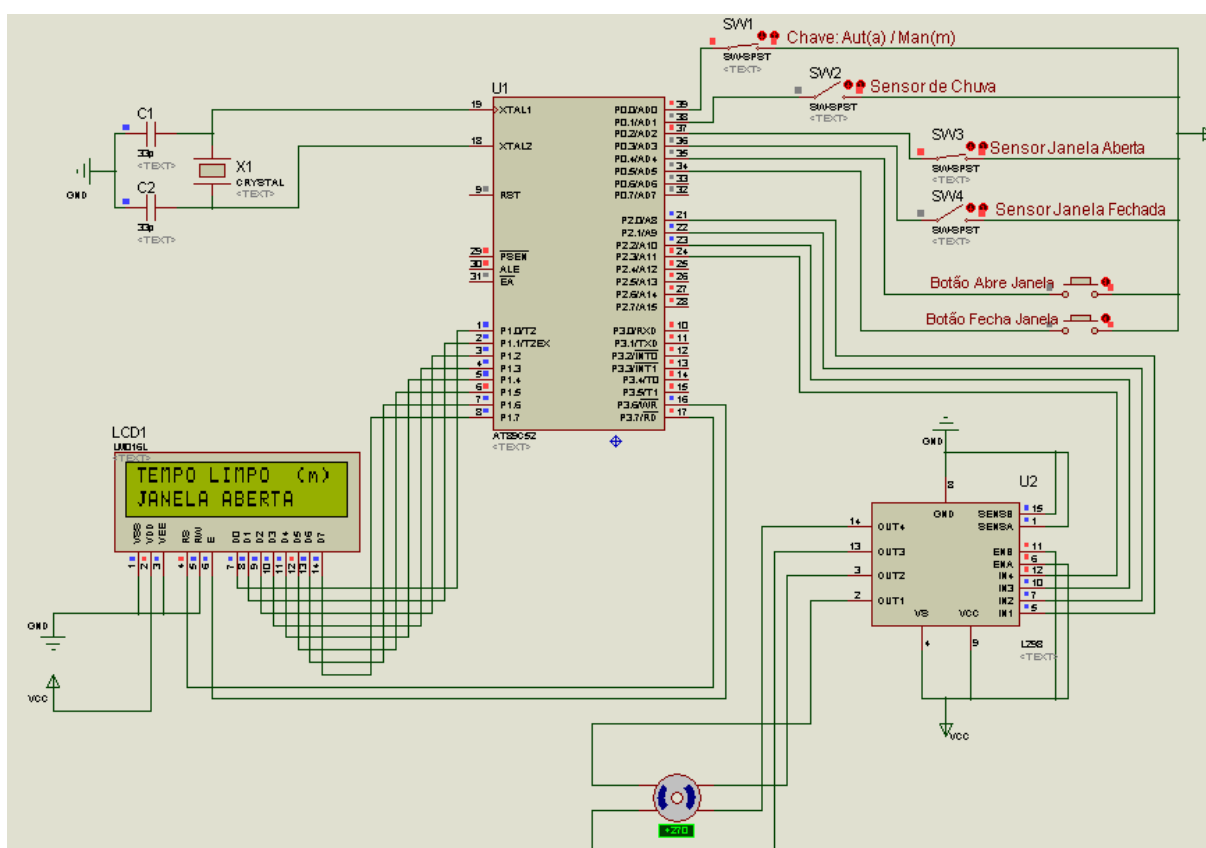


Figura 26 – LCD - Tempo Limpo (m); Janela Aberta

(Autor)

Como já foi dito, quando o sistema estiver operando no modo manual, o usuário tem uma maior interatividade com o sistema, podendo abrir e fechar a janela conforme seu desejo.

A Figura 26 mostra uma situação na qual o sistema se encontrava conforme Figura 25, e após o botão para fechar a janela ser pressionando, apesar de não estar chovendo, a janela se fechou. Nestas condições o display mostra em sua primeira linha “TEMPO LIMPO (m)”, mostrando que não está chovendo e o modo de operação é o manual, e mostra na segunda linha “JANELA FECHADA”, pois o sensor de janela fechada está ativo.

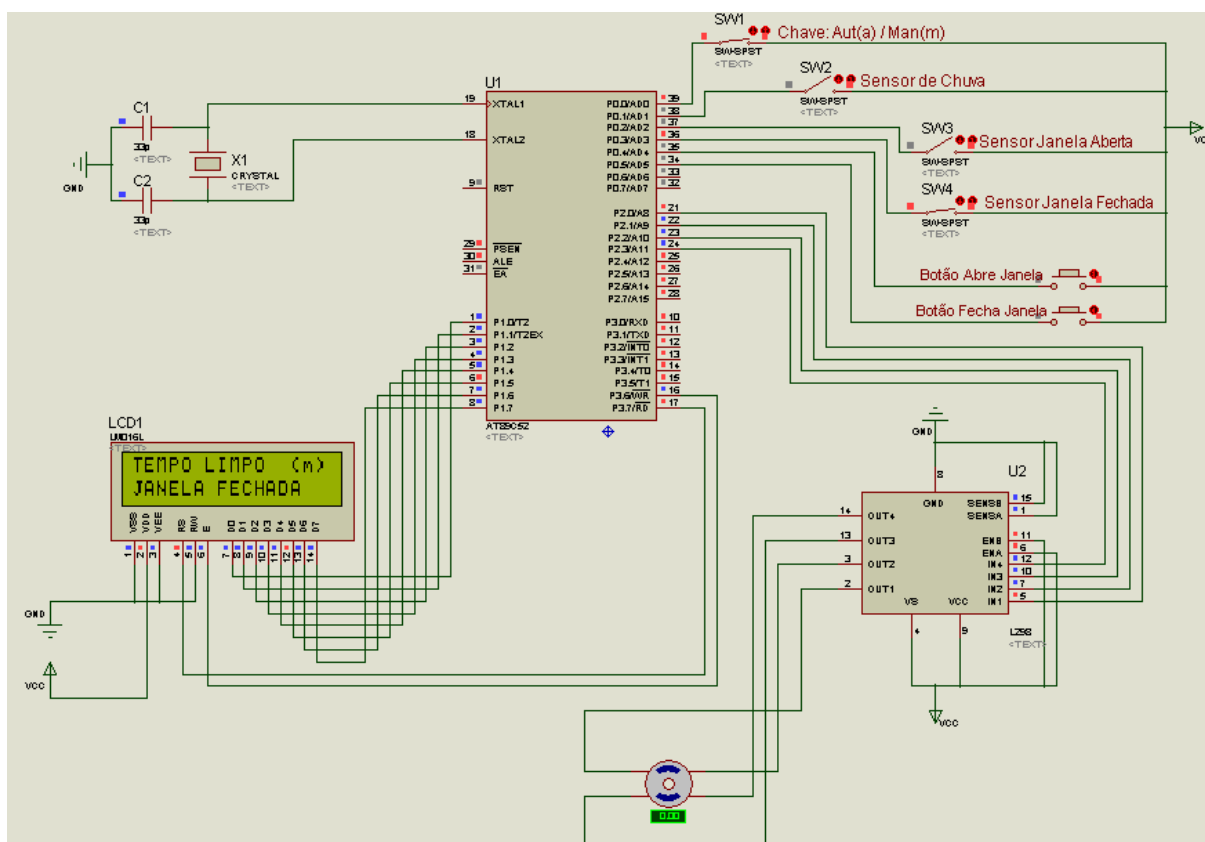


Figura 27 – LCD - Tempo Limpo (m); Janela Fechada

N(Autor)

3.7. – Apresentação do Protótipo

O protótipo começou a ser implementado em uma *protoboard*, visando evitar problemas futuros em um circuito fixo. A Figura 28 mostra como foram feitos os primeiros testes do protótipo.

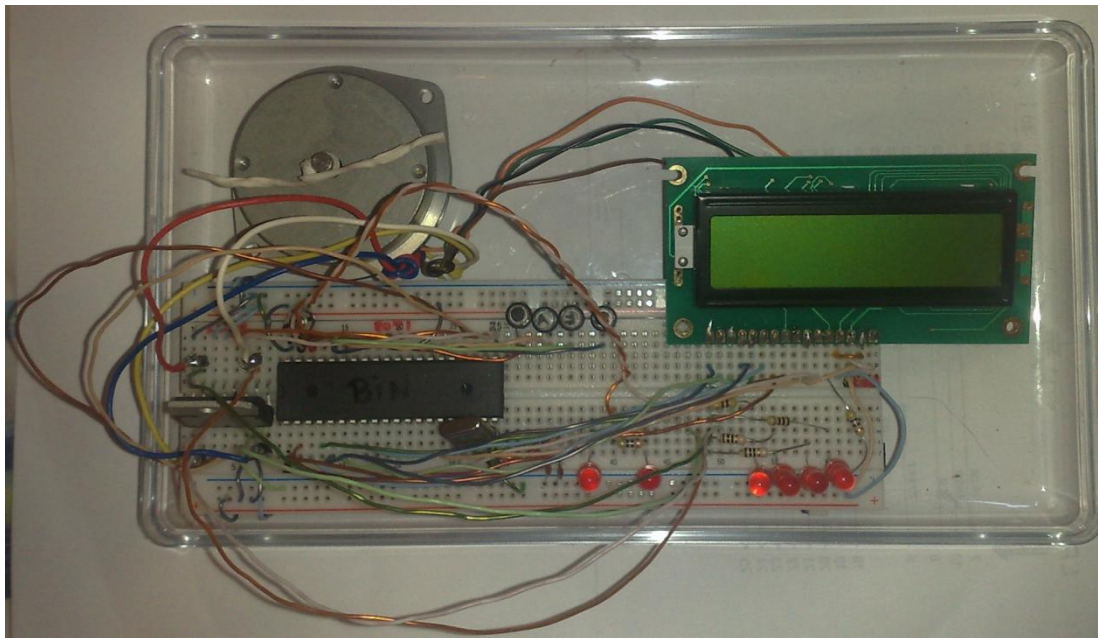


Figura 28 - Foto do Protótipo (1)

(Autor)

Enquanto o protótipo se encontrava nestas condições, os sensores e os botões ainda não havia sido colocados de forma definitiva, sendo que os testes foram feitos com pequenos botões, simulando todos os sensores e todos os botões do protótipo.

Após a constatação de que o circuito estava funcionando de forma satisfatória, o circuito eletrônico foi soldado em uma placa ilhada, conforme pode-se observar na Figura 29.

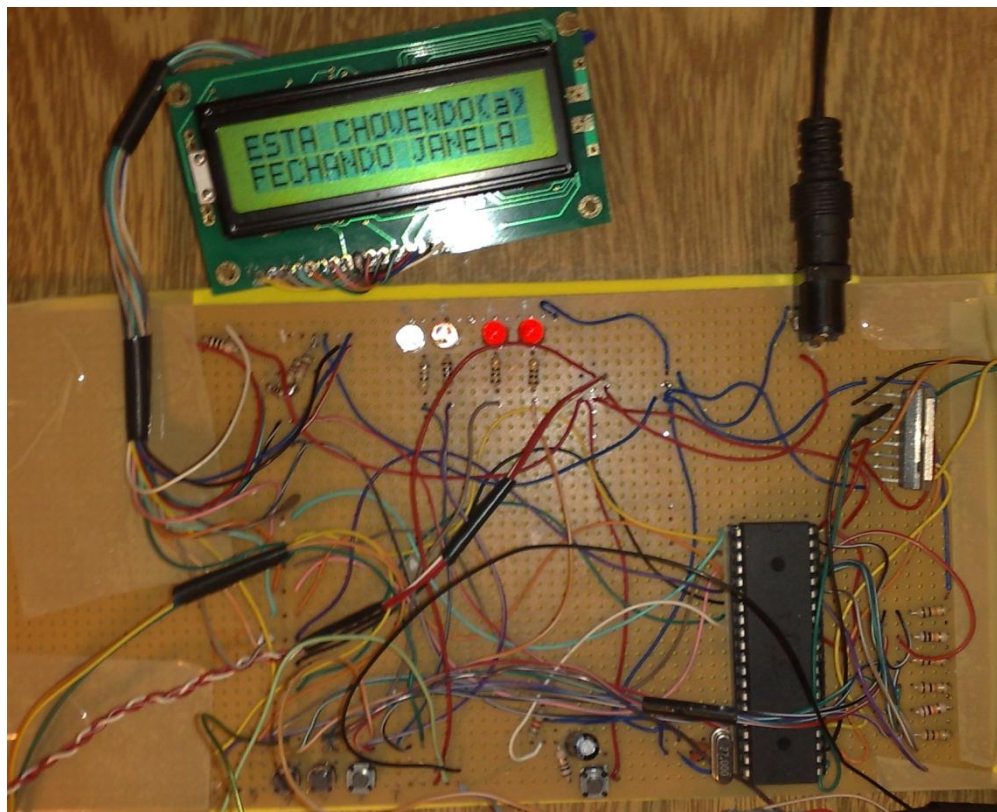


Figura 29 - Foto do Protótipo (2)

(Autor)

Nos primeiros testes com esta placa, ainda não haviam sido inseridos os sensores, botões e chave do circuito.

Com o circuito funcionando já na placa ilhada, a maquete da janela começou a ser preparada, para uma estética melhor foi elaborada uma maquete de um cômodo com uma janela, sendo que a estrutura do cômodo foi feita em isopor, a janela foi feita de acrílico, e a estrutura da janela foi feita em plástico com pequenos trilhos. A base de toda a maquete é uma tábua de madeira.

Pode-se observar na Figura 30 uma visão geral da maquete. O display foi posicionado no teto do cômodo para uma melhor visualização.



Figura 30 - Foto do Protótipo (3)

(Autor)

Nesta figura já é possível verificar o sensor de chuva na parte inferior à direita e o controle à direita, onde está localizado a chave para colocar o protótipo no modo manual ou automático e os botões para abrir e fechar a janela.

A Figura 31 é uma foto onde a visualização dos sensores e outros controles estão melhor.



Figura 31 - Foto do Protótipo (4)

(Autor)

Nesta figura é possível observar os sensores de janela aberta e fechada, eles foram construídos de forma bem simplificada. O princípio de seu funcionamento é bem simples, dois fios desencapados são colocados de forma paralela nas extremidades da janela, sendo que um fio está ligado a fonte de alimentação e o outro fio está ligado ao microcontrolador na porta correspondente ao sensor de janela fechada ou aberta, a janela conta com um papel alumínio colada na suas extremidades, de forma que quando a janela esteja fecha ou aberta, este papel alumínio feche o curto, encaminhando o sinal de janela fecha ou aberta ao microcontrolador.

A ligação física entre o motor de passo e a janela, se dá através de uma pequena linha, que está fixada nas duas extremidades da janela e ao motor de passo. Alguns ganchos foram colados a maquete a fim de funcionarem como roldanas, facilitando o trabalho do motor de passo, conforme pode ser verificado na Figura 32.

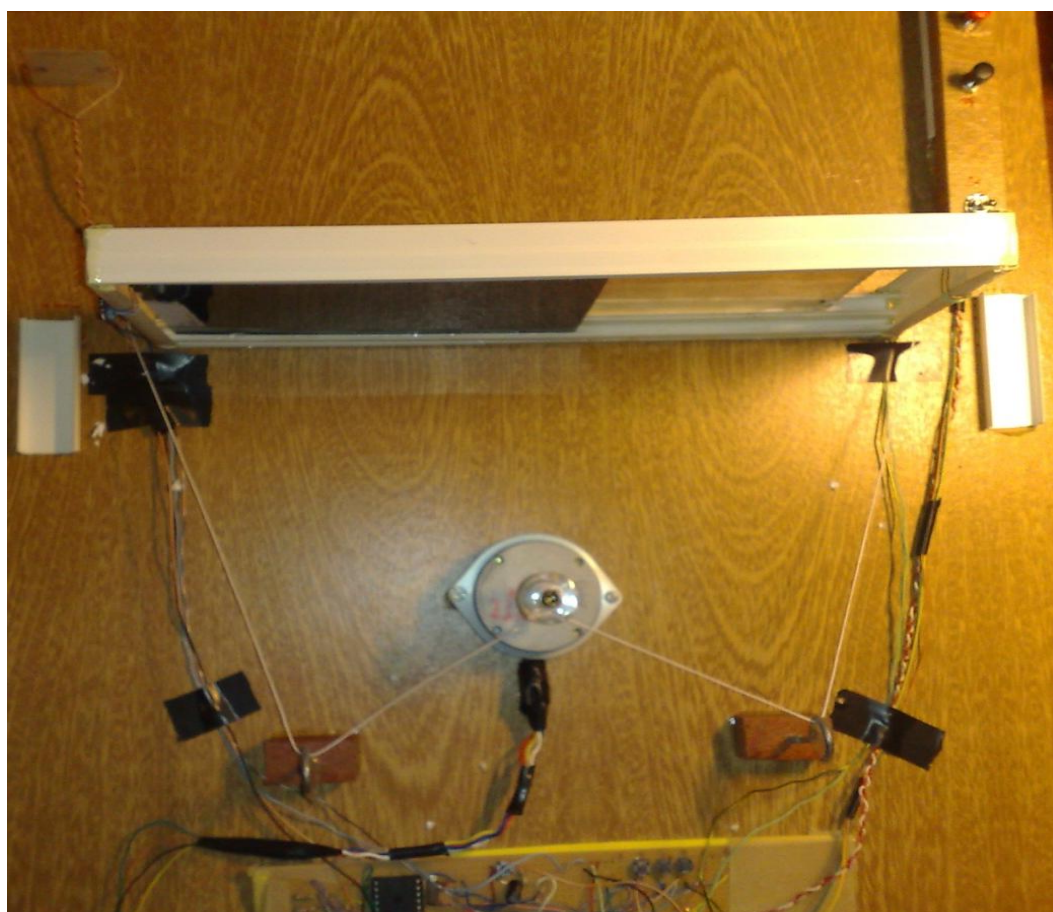


Figura 32 - Foto do Protótipo (5)

(Autor)

A Figura 33 mostra uma foto geral do circuito com todos os sensores ligados e o motor de passo ligado à janela de forma que ela possa abrir e fechar.



Figura 33 - Foto do Protótipo (6)

(Autor)

Nesta figura é possível observar pelo display que o tempo está limpo e a janela está aberta, também podemos observar que o protótipo está operando no modo automático neste momento.

A Figura 34 mostra uma visão frontal do protótipo, nesta foto o projeto está completo e em pleno funcionamento, todos os sensores estão funcionando

plenamente. A estrutura que simula um cômodo pode ser retirada facilmente, expondo assim o circuito eletrônico.



Figura 34 - Foto do Protótipo (7)

(Autor)

No teto do cômodo encontra-se o display, porém, como o cômodo pode ser removido, o display não é fixo. Foi feito um furo no teto para que assim a estrutura do cômodo possa ser retirada sem danificar as ligações do circuito.

CAPÍTULO 4 – CONSIDERAÇÕES FINAIS

4.1. – Dificuldades encontradas

Para a realização deste projeto, várias dificuldades tiveram que ser superadas. Primeiramente a aquisição de um motor de passo que atendesse as especificações do projeto, não foi possível no mercado local e sua aquisição se deu através da internet.

Após a compra do motor, a segunda dificuldade foi encontrar um *driver* que fizesse o motor funcionar. Então depois de várias tentativas com diversos *drivers*, foi utilizado com êxito o circuito integrado L298N.

A aquisição de um sensor de chuva foi ainda mais complicada. Não foi encontrado em sites da internet algum sensor de chuva que não tivesse um preço exorbitante, os preços variaram entre 300 e 800 reais. Após este acontecido, surgiu a idéia de construir um sensor de chuva de forma artesanal, que se mostrou muito satisfatório e de baixíssimo custo.

Uma dificuldade que demandou bastante tempo foi à construção de um circuito para gravação do programa no microcontrolador. Foram três circuitos construídos em vão, desprendendo muito tempo e dinheiro. Com isso, foi comprado uma gravadora já pronta, aumentando o custo do projeto em 200 reais.

O custo total do projeto ficou em torno de 400 reais, porém, caso não fosse necessário comprar uma gravadora para o microcontrolador este preço cairia pela metade.

4.2. – Avaliação Global do Projeto

Este projeto tem uma larga utilização no mercado, é bem possível que em pouco tempo várias residências estejam automatizadas, já existe uma tendência em automatizar as residências em construção.

Faz-se necessário observar que, para implementação deste projeto em janelas reais, o motor de passo utilizado tem ter uma potência maior que o motor utilizado no protótipo, e com isso, o valor final do produto encarece consideravelmente.

4.3. – Conclusão

A automação residencial é o objetivo deste projeto, sendo que a área desenvolvida foi no controle das janela perante precipitação de chuva. Para que fosse possível a realização deste, mostrou-se necessário o desenvolvimento de um sistema composto por um *hardware* e um *software*, para que assim a janela fosse automatizada da forma proposta.

Para a implementação deste projeto, foi utilizada como elemento principal, um microcontrolador da família 8051 responsável por interpretar e aplicar as informações recolhidas pelo sistema, ele é quem controla as informações escritas em um display de cristal líquido, e também é quem controla as informações enviadas para movimentação do motor de passo, porém um circuito integrado, L298N, é quem faz a ligação entre o microcontrolador e o motor de passo.

O sistema se mostrou satisfatório no intuito de automatizar uma janela, sendo que ele ainda permite trabalhar em modo manual e modo automático, ou seja, o usuário pode ter controle sobre a janela, podendo abrir ou fechar, porém esta função só funcionará enquanto não estiver chovendo, pois a janela não pode abrir com a presença de chuva.

O sistema de uma forma geral se mostrou muito útil, podendo ser implementados também em outros ambientes que não sejam residenciais, trazendo mais comodidade e conforto para as pessoas ou as empresas.

4.4. – Sugestões para trabalhos futuros

Como sugestão para outros projetos nesta área, seria a inserção de vários outros sensores nas janelas, como sensor de intensidade de luz, fechando persianas ou cortinas, sensor de vento, fechando a janela e evitando possíveis danos, sensor de movimento, no intuito de fechar a janela caso algum intruso se aproxime.

Outra boa sugestão seria a automação da residência por completo da casa, tendo uma central única onde seria possível ver a situação de cada ponto automatizado. Esta central também poderia ter um suporte *on line*, onde o usuário teria o controle das automações via *web*.

REFERENCIAS BIBLIOGRAFICAS

- [FITZGERALD, 2006] FITZGERALD, A. E.; Charles Kingsley Jr.; Stephen D. Umans. **Máquinas Elétricas**. 6ª ed. Porto Alegre: Bookman, 2006.
- [GIMENEZ, 2002] GIMENEZ, Salvador Pinillos. **Microcontroladores 8051**: Teoria do hardware e do software / Aplicações em controle digital / Laboratório e simulação. São Paulo: Pearson Education do Brasil Ltda, 2002.
- [MARINHO, 2001] EDINALDO DOS SANTOS; MARINHO, JOSE EDSON DOS SANTOS. “**Mini curso de Microcontrolador**”. Saber Eletrônica Especia nº2, Janeiro 2001.
- [NICOLOSI, 2000] NICOLOSI, Denys Emílio Campion. **Microcontrolador 8051 Detalhado**. São Paulo: Érica, 2000.
- [SICA, 2006] SICA, Carlos. **Sistemas Automáticos com Microcontroladores 8031/8051**. São Paulo: Novatec, 2006.
- [TIPLER, 2006] TIPLER, Paul A. **Física, Volume 2: Eletricidade e Magnetismo, Ótica**. Rio de Janeiro: LTC, 2006.
- [TORO, 1994] TORO, Vincent Del. **Fundamentos de Máquinas Elétricas**. Rio de Janeiro: LTC, 1994.
- [YOUNG, 2004] YOUNG, Hugh D. **Física III, Eletromagnetismo**. São Paulo: Addison Wesley, 2004.

- [BRITES, 2008] BRITES, Felipe Gonçalves; Projeto Acadêmico pela Universidade Federal Fluminense. **Motor de Passo**; julho / 2008. Disponível em (agosto 2009):

<http://www.telecom.uff.br/pet/petws/downloads/tutoriais/stepmotor/stepmotor2k81119.pdf>

- [MOHR, 2004] MOHR, Hari Bruno : Werner Kraus Jr. : Raimes Moraes; Artigo Acadêmico; Universidade Federal de Santa Catarina; **A família de microcontroladores 8051**. janeiro/2004. Disponível em (agosto 2009):

<http://www.das.ufsc.br/~werner/eel7030/8051/Apostila8051Hari.pdf>

- [NUNES, 2009] NUNES, Paulo; Artigo: **Conceito de Fluxograma**. Abril/2009. Disponível em (novembro 2009):

<http://www.knoow.net/cienceconempr/gestao/fluxograma.htm>

- [TERROSO, 2009] TERROSO, Anderson R.; Pontifícia Universidade Católica do Rio Grande do Sul – PUCRS; **Exercício sobre display LCD**. Disponível em (agosto 2009):

http://www.ee.pucrs.br/~terroso/PROGRAMA2_C_LCD_DA_PLACA_2007_2.pdf

- [UNIMAR] Universidade de Marília – UNIMAR ; **Trabalho sobre eletroímã**. Disponível em (novembro 2009):

http://www.profcarlos.net/index.php?option=com_docman&task=doc_download&gid=37&Itemid=5

- [PADUA, 2004] Helcias Bernardo de Pádua; **Série: ÁGUA – parte XII**; setembro/2004. Disponível em (novembro 2009):

<http://www.fernandosantiago.com.br/aguadura.htm>

APÊNDICE

Programa inserido no microcontrolador

```
#include "8051.h"
```

```
#define dado P1 //declarando porta P1 para envio de dados ao LCD
```

```
#define rs P3_7 //declarando os sinais P3.7 ao sinal RS
```

```
#define en P3_6 //declarando os sinais P3.6 ao sinal EN
```

```
#define led1 P2_0 // declara porta para controlar passo 1 do motor
```

```
#define led2 P2_1 // declara porta para controlar passo 2 do motor
```

```
#define led3 P2_2 // declara porta para controlar passo 3 do motor
```

```
#define led4 P2_3 // declara porta para controlar passo 4 do motor
```

```
#define bot1 P0_0 // modo manual / automatico
```

```
#define bot2 P0_1 // sensor de chuva
```

```
#define bot3 P0_2 // sensor de janela aberta
```

```
#define bot4 P0_3 // sensor de janela fechada
```

```
#define bot5 P0_4 // botão abre janela
```

```
#define bot6 P0_5 // botão fecha janela
```

```
void config_lcd(void); //função configura lcd
```

```
void linha1(void); //função escreve linha 1 do lcd
```

```
void linha2(void); //função escreve linha 2 do lcd
```

```
void escreve_char(char msn[]); //função escreve string
```

```
void delay_6us(void); //função de delay de 6 microsegundos
```

```
void delay_6ms(void); //função de delay de 8 milisegundos
```

```
void wr_cmd(void); //função para envio de configuração ao lcd
```

```
void wr_char(void); //função para envio de string ao lcd
```

```

void tempo(void);                //função de intervalo de tempo entre passos do motor
void tempo_limpo_abre_janela(void); //função para abrir janela enquanto o tempo estiver limpo
void chuva_fecha_janela(void);    //função para fechar janela enquanto estiver chovendo
void tempo_limpo_fecha_janela(void); //função para fechar janela enquanto o tempo estiver limpo
void status_lcd(void);            //função para escrever no lcd o status da janela

```

```

int x1, x2, x3, x4;

```

```

//*****

```

```

void main(void)
{
    config_lcd();
    status_lcd();
    while(1)
    {
        x1=100;
        while(bot1==0 && bot2==0)
        {
            if(x1==100)
            {
                tempo_limpo_abre_janela();
                x1=0;
            }
        }
        x2=100;
        while(bot1==0 && bot2==1)
    }
}

```

```

{
    if(x2==100)
    {
        chuva_fecha_janela();

        x2=0;
    }
}

x3=100;

while(bot1==1 && bot2==1)
{
    if(x3==100)
    {
        chuva_fecha_janela();

        x3=0;
    }
}

x4=100;

while(bot1==1 && bot2==0)
{
    if(bot5==1 || bot6==1)
    {
        if(bot5==1)
        {
            tempo_limpo_abre_janela();
        }

        if(bot6==1)
        {
            tempo_limpo_fecha_janela();

```



```

void linha1(void)                // define o local de inicio de impressao na linha 1 do LCD

{

    dado = 0x80;

    wr_cmd();

}

//*****

void linha2(void)                // define o local de inicio de impressao na linha 2 do LCD

{

    dado = 0xc0;

    wr_cmd();

}

//*****

void config_lcd(void)            // função de comandos para configuração do Display

{

    dado = 0x38;

    wr_cmd();

    dado = 0x07;

    wr_cmd();

    dado = 0x0E;

    wr_cmd();

    dado = 0x06;

    wr_cmd();

```

```

    dado = 0x01;

    wr_cmd();

}

//*****

void delay_6us(void)          //rotina de delay de 2 microsegundos
{
    _asm                    //inicio de trecho assembly

        0$: mov  r0, #6
        1$: djnz r0, 1$

    _endasm;                //fim do trecho assembly

    // 1 ciclo com 12Mhz = 1us => 1 ciclo com 27Mhz = 0.444444444444 microsegundos
    // #ciclos = 1c+(2c)*6
    // #ciclos = 13
    // #tempo = 13 * 0.444444444444 = 5.777 microsegundos
}

//*****

void delay_6ms(void)          //rotina de delay de 2 milisegundos
{
    _asm                    //inicio de trecho assembly

        1$: mov  r1, #27      //1c
        2$: mov  r2, #255     //1c
        3$: djnz r2, 3$       //2c

```

```

        djnz r1, 2$           //2c

_endasm;                      //fim do trecho assembly


// 1 ciclo com 12Mhz = 1us => 1 ciclo com 27Mhz = 0.444444444444 microsegundos
// #ciclos = 1c+[1c+(2c)*255+2c]*27
// #ciclos = 13852
// tempo  = 13852 * 0.444444444444 = 6.156 milisegundos
}


//*****

void wr_char()                // função escreve string no Display
{
    rs = 1;
    en = 1;
    delay_6us();
    en = 0;
    delay_6ms();
}


//*****

void wr_cmd(void)             // função envia comando ao Display
{
    rs = 0;
    en = 1;
    delay_6us();
    en = 0;

```



```

    delay_6ms();

}

//*****

void tempo(void)          // Função para controle de tempo entre um passo e outro
{ int d;

    for(d=0;d<4;d++)

    {

        delay_6ms();

    }

}

//*****

void tempo_limpo_abre_janela(void)
{

    int i;

    if(bot3==0) //sensor de janela aberta
    {

        if(bot1==1)
        {

            linha1();

            escreve_char("TEMPO LIMPO (m)   ");

            linha2();

            escreve_char("ABRINDO JANELA   ");

        }

    }
}

```

```

else
{
    linha1();

    escreve_char("TEMPO LIMPO (a)  ");

    linha2();

    escreve_char("ABRINDO JANELA  ");
}
}

while(bot3==0)
{
    for(i=1;i<5;i++)
    {
        tempo();

        switch(i)
        {
            case 1:

                led1=1;

                led2=0;

                led3=0;

                led4=0;

                break;

            case 2:

                led1=0;

                led2=1;

                led3=0;

                led4=0;

                break;

            case 3:

```

```

    led1=0;

    led2=0;

    led3=1;

    led4=0;

    break;

    case 4:

    led1=0;

    led2=0;

    led3=0;

    led4=1;

    break;

    default:

    break;

    }

    }

    }

    status_lcd();

}

//*****

void chuva_fecha_janela(void)

{

    int i;

    if(bot4==0) //sensor de janela fechada

    {

        if(bot1==1)

```

```

{
    linha1();

    escreve_char("ESTA CHOVENDO(m)   ");

    linha2();

    escreve_char("FECHANDO JANELA   ");
}

else

{
    linha1();

    escreve_char("ESTA CHOVENDO(a)   ");

    linha2();

    escreve_char("FECHANDO JANELA   ");
}
}

while(bot4==0)

{
    for(i=4;i>0;i--)

    {
        tempo();

        switch(i)

        {
            case 1:

                led1=1;

                led2=0;

                led3=0;

                led4=0;

                break;

            case 2:

```

```

    led1=0;

    led2=1;

    led3=0;

    led4=0;

    break;

    case 3:

    led1=0;

    led2=0;

    led3=1;

    led4=0;

    break;

    case 4:

    led1=0;

    led2=0;

    led3=0;

    led4=1;

    break;

    default:

    break;

}

}

}

status_lcd();

}

//*****

void tempo_limpo_fecha_janela(void)

{

```

```

int i;

if(bot4==0) //sensor de janela fechada
{
    if(bot1==1)
    {
        linha1();

        escreve_char("TEMPO LIMPO (m)   ");

        linha2();

        escreve_char("FECHANDO JANELA   ");
    }
    else
    {
        linha1();

        escreve_char("TEMPO LIMPO (a)   ");

        linha2();

        escreve_char("FECHANDO JANELA   ");
    }
}

while(bot4==0)
{
    for(i=1;i<5;i++)
    {
        tempo();

        switch(i)
        {
            case 1:

                led1=1;

```

```
    led2=0;

    led3=0;

    led4=0;

    break;

    case 2:

    led1=0;

    led2=1;

    led3=0;

    led4=0;

    break;

    case 3:

    led1=0;

    led2=0;

    led3=1;

    led4=0;

    break;

    case 4:

    led1=0;

    led2=0;

    led3=0;

    led4=1;

    break;

    default:

    break;

}

}

}

status_lcd();
```

```
}
```

```
/*******
```

```
void status_lcd(void)
```

```
{
```

```
    if(bot1==0)
```

```
    {
```

```
        if(bot2==0 && bot4==1)
```

```
        {
```

```
            linha1();
```

```
            escreve_char("TEMPO LIMPO (a)  ");
```

```
            linha2();
```

```
            escreve_char("JANELA FECHADA  ");
```

```
        }
```

```
        if(bot2==0 && bot3==1)
```

```
        {
```

```
            linha1();
```

```
            escreve_char("TEMPO LIMPO (a)  ");
```

```
            linha2();
```

```
            escreve_char("JANELA ABERTA  ");
```

```
        }
```

```
        if(bot2==1 && bot4==1)
```

```
        {
```

```
            linha1();
```

```
            escreve_char("ESTA CHOVENDO(a)  ");
```

```
            linha2();
```

```
            escreve_char("JANELA FECHADA  ");
```

```
        }
```



```

}

else

{

    if(bot2==0 && bot4==1)

    {

        linha1();

        escreve_char("TEMPO LIMPO (m)  ");

        linha2();

        escreve_char("JANELA FECHADA  ");

    }

    if(bot2==0 && bot3==1)

    {

        linha1();

        escreve_char("TEMPO LIMPO (m)  ");

        linha2();

        escreve_char("JANELA ABERTA  ");

    }

    if(bot2==1 && bot4==1)

    {

        linha1();

        escreve_char("ESTA CHOVENDO(m)  ");

        linha2();

        escreve_char("JANELA FECHADA  ");

    }

}

}

}

```

```

//*****

```

ANEXO I - DATASHEET DO CI L298N



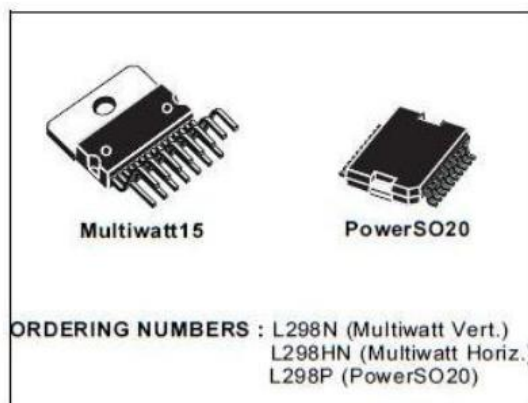
L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

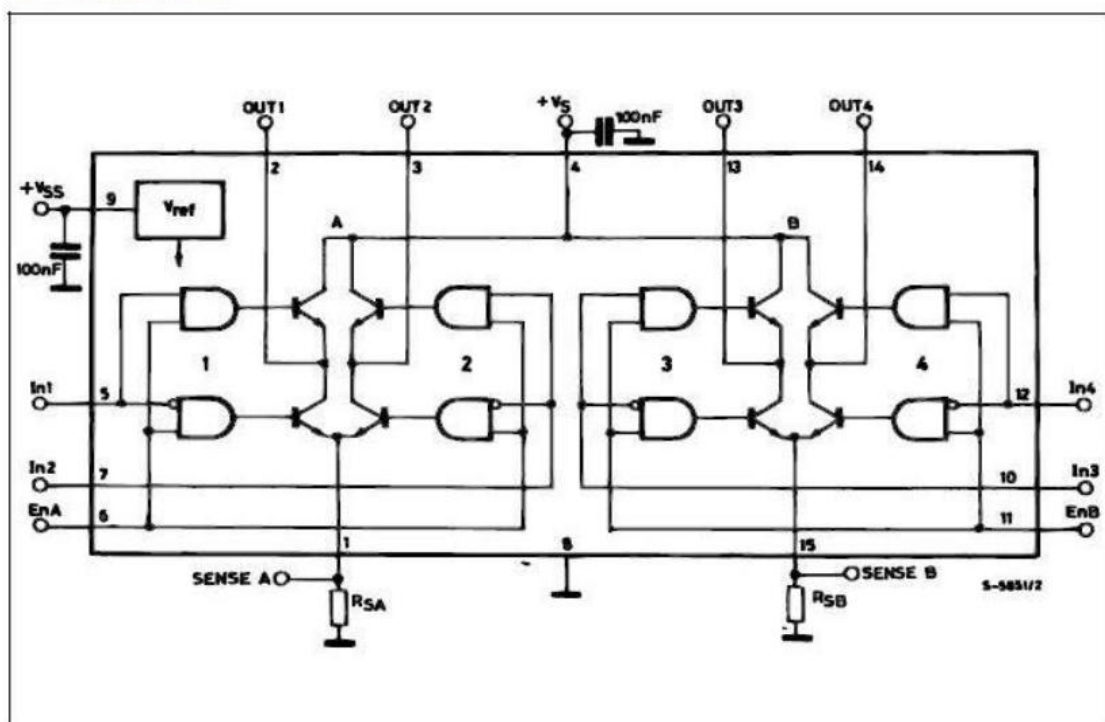
DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-



nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

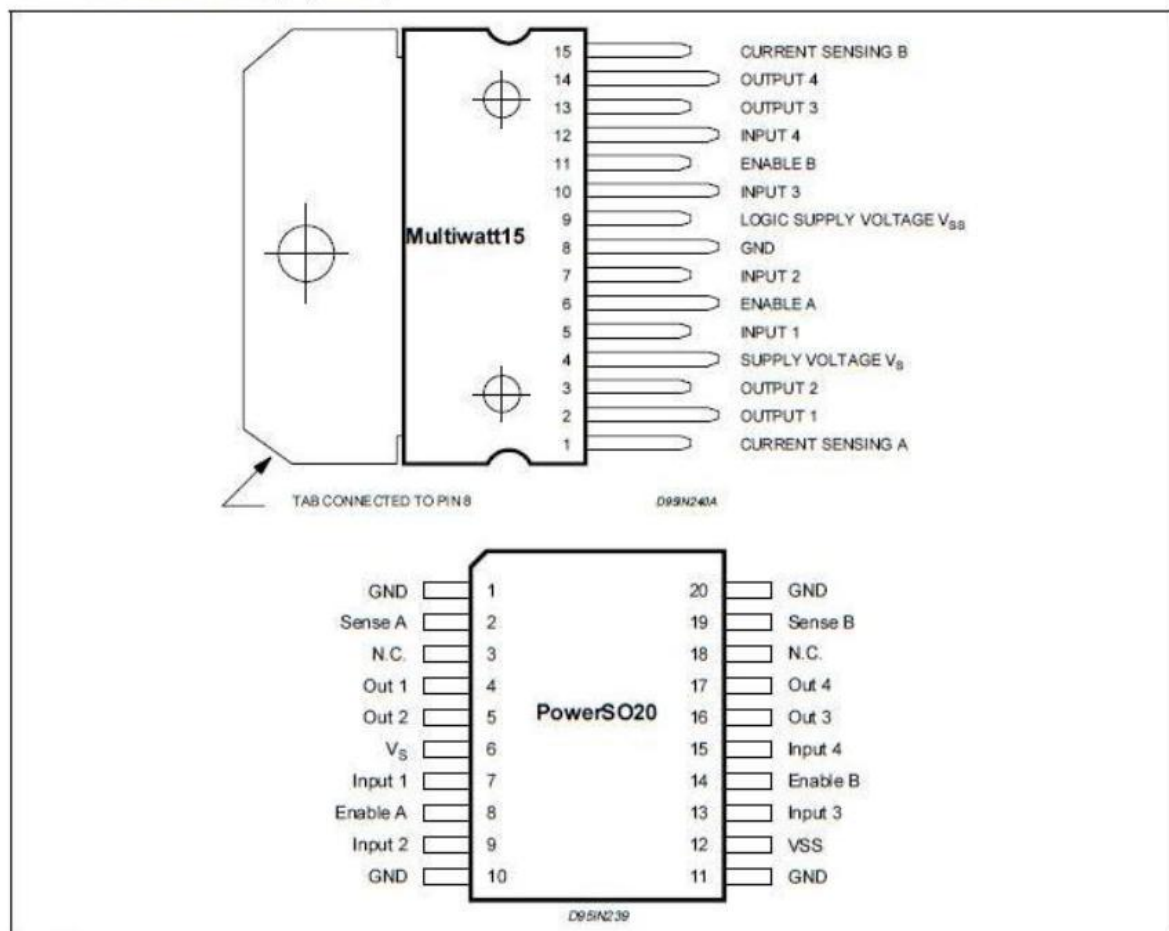
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i, V_{en}	Input and Enable Voltage	-0.3 to 7	V
I_O	Peak Output Current (each Channel)		
	- Non Repetitive ($t = 100\mu s$)	3	A
	- Repetitive (80% on -20% off; $t_{on} = 10ms$)	2.5	A
	- DC Operation	2	A
V_{sens}	Sensing Voltage	-1 to 2.3	V
P_{tot}	Total Power Dissipation ($T_{case} = 75^\circ C$)	25	W
T_{op}	Junction Operating Temperature	-25 to 130	$^\circ C$
T_{stg}, T_J	Storage and Junction Temperature	-40 to 150	$^\circ C$

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter	PowerSO20	Multiwatt15	Unit
$R_{\theta j-case}$	Thermal Resistance Junction-case	Max.	3	$^\circ C/W$
$R_{\theta j-amb}$	Thermal Resistance Junction-ambient	Max.	35	$^\circ C/W$

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V _S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A 100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
—	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_J = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V _S	Supply Voltage (pin 4)	Operative Condition	V _{IH} +2.5		46	V
V _{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I _S	Quiescent Supply Current (pin 4)	V _{en} = H; I _L = 0 V _I = L V _I = H		13 50	22 70	mA mA
		V _{en} = L V _I = X			4	mA
I _{SS}	Quiescent Current from V _{SS} (pin 9)	V _{en} = H; I _L = 0 V _I = L V _I = H		24 7	36 12	mA mA
		V _{en} = L V _I = X			6	mA
V _{IL}	Input Low Voltage (pins 5, 7, 10, 12)		-0.3		1.5	V
V _{IH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		V _{SS}	V
I _{IL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V _I = L			-10	μA
I _{IH}	High Voltage Input Current (pins 5, 7, 10, 12)	V _I = H ≤ V _{SS} -0.6V		30	100	μA
V _{en} = L	Enable Low Voltage (pins 6, 11)		-0.3		1.5	V
V _{en} = H	Enable High Voltage (pins 6, 11)		2.3		V _{SS}	V
I _{en} = L	Low Voltage Enable Current (pins 6, 11)	V _{en} = L			-10	μA
I _{en} = H	High Voltage Enable Current (pins 6, 11)	V _{en} = H ≤ V _{SS} -0.6V		30	100	μA
V _{CEsat} (H)	Source Saturation Voltage	I _L = 1A	0.95	1.35	1.7	V
		I _L = 2A		2	2.7	V
V _{CEsat} (L)	Sink Saturation Voltage	I _L = 1A (5)	0.85	1.2	1.6	V
		I _L = 2A (5)		1.7	2.3	V
V _{CEsat}	Total Drop	I _L = 1A (5)	1.80		3.2	V
		I _L = 2A (5)			4.9	V
V _{sens}	Sensing Voltage (pins 1, 15)		-1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V)	Source Current Turn-off Delay	$0.5 V_I$ to $0.9 I_L$ (2); (4)		1.5		μs
T_2 (V)	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		0.2		μs
T_3 (V)	Source Current Turn-on Delay	$0.5 V_I$ to $0.1 I_L$ (2); (4)		2		μs
T_4 (V)	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.7		μs
T_5 (V)	Sink Current Turn-off Delay	$0.5 V_I$ to $0.9 I_L$ (3); (4)		0.7		μs
T_6 (V)	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.25		μs
T_7 (V)	Sink Current Turn-on Delay	$0.5 V_I$ to $0.9 I_L$ (3); (4)		1.6		μs
T_8 (V)	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.2		μs
f_c (V)	Commutation Frequency	$I_L = 2A$		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (2); (4)		3		μs
T_2 (V_{en})	Source Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (2); (4)		1		μs
T_3 (V_{en})	Source Current Turn-on Delay	$0.5 V_{en}$ to $0.1 I_L$ (2); (4)		0.3		μs
T_4 (V_{en})	Source Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (2); (4)		0.4		μs
T_5 (V_{en})	Sink Current Turn-off Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		2.2		μs
T_6 (V_{en})	Sink Current Fall Time	$0.9 I_L$ to $0.1 I_L$ (3); (4)		0.35		μs
T_7 (V_{en})	Sink Current Turn-on Delay	$0.5 V_{en}$ to $0.9 I_L$ (3); (4)		0.25		μs
T_8 (V_{en})	Sink Current Rise Time	$0.1 I_L$ to $0.9 I_L$ (3); (4)		0.1		μs

1) Sensing voltage can be $-1 V$ for $t \leq 50 \mu s$; in steady state $V_{sens} \min \geq -0.5 V$.

2) See fig. 2.

3) See fig. 4.

4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current

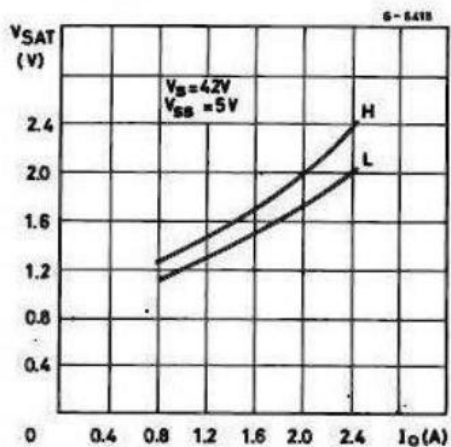
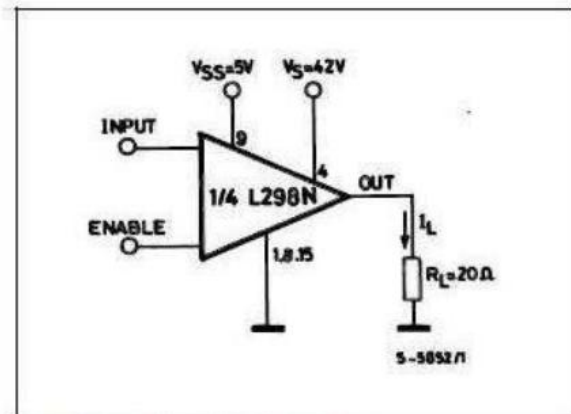


Figure 2 : Switching Times Test Circuits.

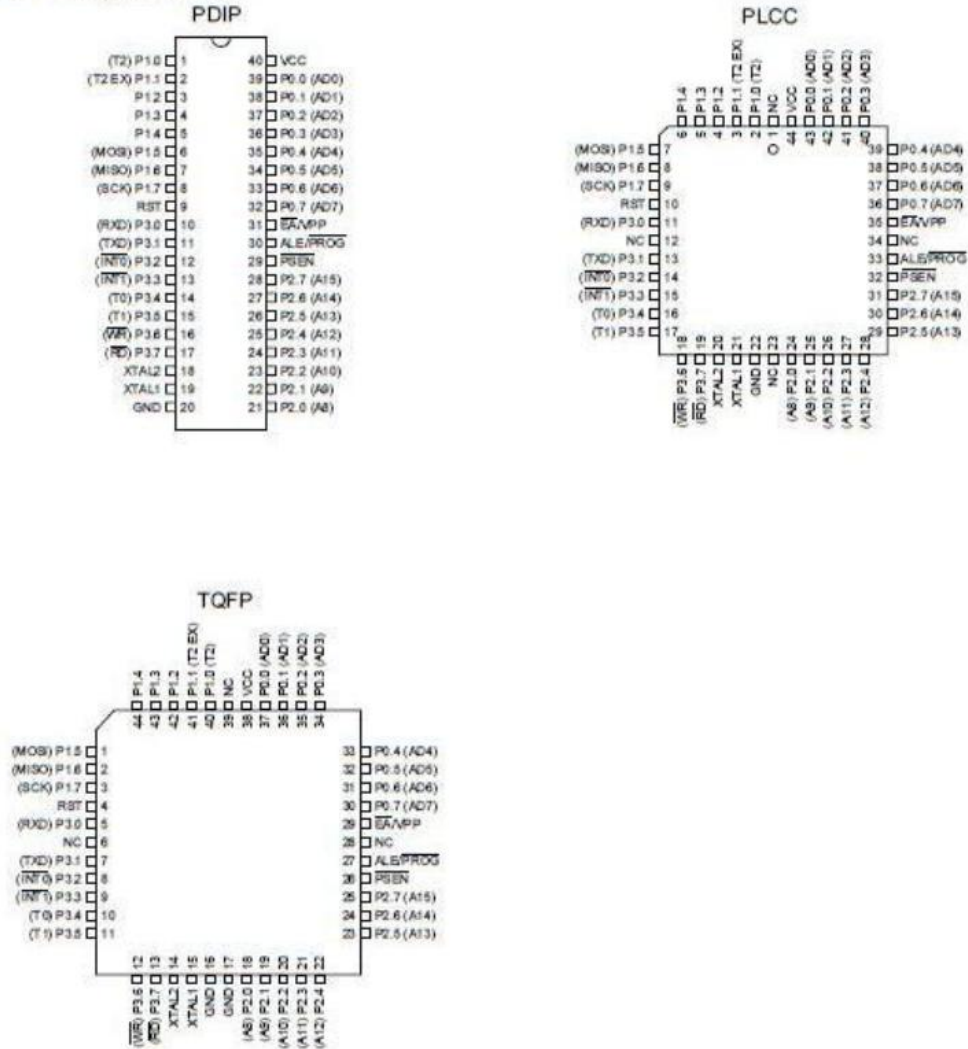


Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

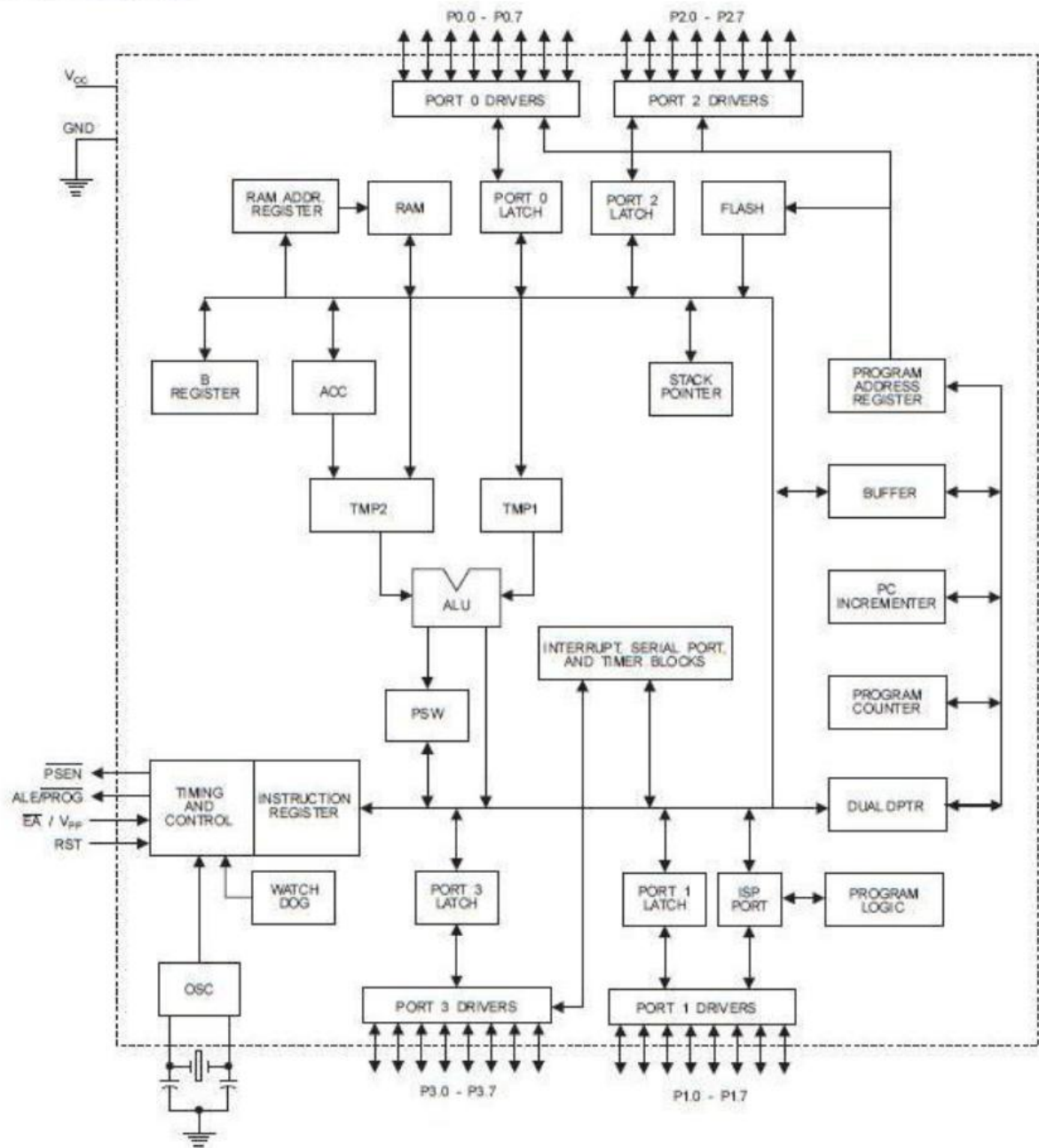
ANEXO II - DATASHEET DO MICROCONTROLADOR AT89S52



Pin Configurations



Block Diagram





Pin Description

VCC

Supply voltage.

GND

Ground.

Port 0

Port 0 is an 8-bit open drain bidirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1

Port 1 is an 8-bit bidirectional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

Port 2

Port 2 is an 8-bit bidirectional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to

external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ R1), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table.

Port 3 also receives some control signals for Flash programming and verification.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	INT0 (external interrupt 0)
P3.3	INT1 (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	WR (external data memory write strobe)
P3.7	RD (external data memory read strobe)

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 96 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

ALE/PROG

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (PROG) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is

weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

PSEN

Program Store Enable ($\overline{\text{PSEN}}$) is the read strobe to external program memory.

When the AT89S52 is executing code from external program memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle, except that two $\overline{\text{PSEN}}$ activations are skipped during each access to external data memory.

EA/VPP

External Access Enable. $\overline{\text{EA}}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH.

Note, however, that if lock bit 1 is programmed, $\overline{\text{EA}}$ will be internally latched on reset.

$\overline{\text{EA}}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

Table 1. AT89S52 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111		AUXR1 XXXXXXXX0				WDTRST XXXXXXXX	0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0	8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	PCON 0XXX0000	87H





Memory Organization

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

Program Memory

If the \overline{EA} pin is connected to GND, all program fetches are directed to external memory.

On the AT89S52, if \overline{EA} is connected to V_{CC} , program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.

Data Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access of the SFR space.

For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV 0A0H, #data
```

Instructions that use indirect addressing access the upper 128 bytes of RAM. For example, the following indirect addressing instruction, where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

```
MOV @R0, #data
```

Note that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space.

Oscillator Characteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 11. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 12. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

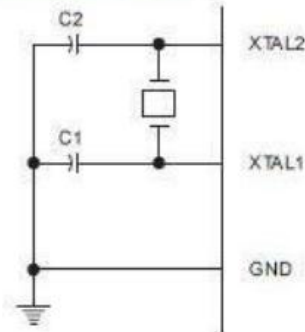
Note that when idle mode is terminated by a hardware reset, the device normally resumes program execution from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.

Power-down Mode

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by an enabled external interrupt. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held

active long enough to allow the oscillator to restart and stabilize.

Figure 11. Oscillator Connections



Note: C1, C2 = 30 pF \pm 10 pF for Crystals
= 40 pF \pm 10 pF for Ceramic Resonators

Figure 12. External Clock Drive Configuration

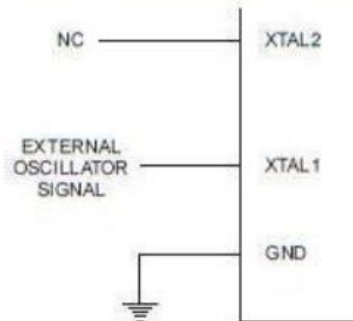


Table 6. Status of External Pins During Idle and Power-down Modes

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

ANEXO III - DATASHEET DO DISPLAY 16X2

MGD1602B SERIES LCD MODULE

1.0 MECHANICAL SPECS

1. Overall Module Size	84.0mm(W) x 44.0mm(H) x max 13.5mm(D) for LED backlight version 84.0mm(W) x 44.0mm(H) x max 9.5mm(D) for reflective version
2. Dot Size	0.56mm(W) x 0.61mm(H)
3. Dot Pitch	0.61mm(W) x 0.66mm(H)
4. Duty	1/16
5. Controller IC	KS0066U or Equivalent
6. LC Fluid Options	TN, STN
7. Polarizer Options	Reflective, Transflective, Transmissive
8. Backlight Options	LED
9. Temperature Range Options	Standard temp.: (Operating: 0°C ~ 50°C; Storage: -20°C ~ 70°C) Wide temp.: (Operating: -20°C ~ 70°C; Storage: -30°C ~ 80°C)

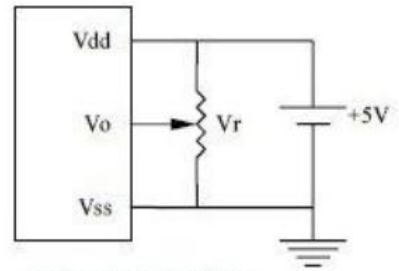
2.0 ABSOLUTE MAXIMUM RATINGS

Item	Symbol	Min	Typ	Max	Unit
Operating temperature (Standard)	Top	0	-	50	°C
Storage temperature (Standard)	Tst	-20	-	70	°C
Operating temperature (Wide temperature)	Top	-20	-	70	°C
Storage temperature (Wide temperature)	Tst	-30	-	80	°C
Input voltage	Vin	Vss	-	Vdd	V
Supply voltage for logic	Vdd- Vss	2.7	-	5.5	V
Supply voltage for LCD drive	Vdd- Vo	3.0	4.5	6.5	V

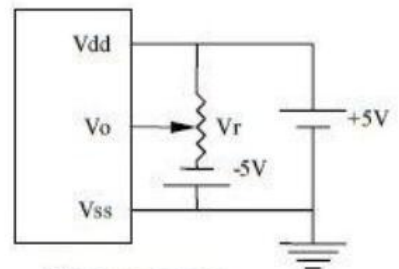
7.0 PIN ASSIGNMENT

Pin No.	Symbol	Function
1	Vss	Ground
2	Vdd	+5V
3	Vo	LCD contrast adjust
4	RS	Register select
5	R/W	Read / write
6	E	Enable
7	DB0	Data bit 0
8	DB1	Data bit 1
9	DB2	Data bit 2
10	DB3	Data bit 3
11	DB4	Data bit 4
12	DB5	Data bit 5
13	DB6	Data bit 6
14	DB7	Data bit 7
+	BL+	Power Supply for BL+
-	BL-	Power Supply for BL-

8.0 POWER SUPPLY



STANDARD TEMP RANGE



WIDE TEMP RANGE

$V_r = 10K\Omega \sim 20K\Omega$

9.0 TIMING CHARACTERISTICS

Item	Symbol	Test Condition	Min.	Typ.	Max.	Unit
Enable cycle time	t_c	Fig. a, Fig. b	500	-	-	ns
Enable pulse width	t_w	Fig. a, Fig. b	220	-	-	ns
Enable rise/fall time	t_r, t_f	Fig. a, Fig. b	-	-	25	ns
RS, R/W set up time	t_{su}	Fig. a, Fig. b	40	-	-	ns
RS, R/W hold time	t_{H}	Fig. a, Fig. b	10	-	-	ns
Data delay time	t_d	Fig. b	-	-	120	ns
Data set up time	t_{dsu}	Fig. a	60	-	-	ns
Data hold time	t_{dH}	Fig. a, Fig. b	20	-	-	ns

